UNIVERSITÉ CÔTE D'AZUR

ÉCOLE DOCTORALE
SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

THÈSE DE DOCTORAT

Raffinement en résolution et précision pour la compression et la simulation de maillages volumiques en géosciences

Lauriane BOUARD

IFP Energies nouvelles — Direction des sciences et technologies numériques
Laboratoire I3S, Sophia Antipolis
CNRS — Centre national de la recherche scientifique

Présentation en vue de l'obtention du grade de docteur en Automatique, Traitement du signal et des Images
de l'Université Côte d'Azur
Dirigée par Marc Antonini
Laurent Duval & Frédéric Payan
Soutenue le 01.03.2021

Devant le jury composé de:
Marc Antonini, CNRS, Université Côte d'Azur, I3S (directeur de thèse)
Amel Benazza-Benyahia, SUP'COM Tunis (rapportrice)
Laure Blanc Feraud, CNRS, Université Côte d'Azur, I3S (présidente)
Guillaume Caumon, ENSG-GeoRessources, Université de Lorraine (rapporteur)
Julie Digne, CNRS, LIRIS, Université Claude Bernard Lyon 1(examinatrice)
Laurent Duval, IFP Energies nouvelles (encadrant)
Peter Lindstrom, Lawrence Livermore National Laboratory (examinateur)
Frédéric Payan, Université Côte d'Azur, I3S, CNRS (encadrant)
Christophe Preux, IFP Energies nouvelles (invité)

Présidente du jury		
Laure Blanc Feraud,	Directrice de Recherche, CNRS,	Université Côte d'Azur
Rapporteurs		
Amel Benazza-Benyahia,	Professeure des Universités,	SUP'COM Tunis
Guillaume Caumon,	Professeur des Universités,	Université de Lorraine
Examinateurs		
Julie Digne,	Chargée de Recherche CNRS,	Université Claude Bernard Lyon 1
Peter Lindstrom,	Chercheur,	Lawrence Livermore National Laboratory
Directeur de thèse		
Marc Antonini,	Directeur de Recherche, CNRS,	Université Côte d'Azur
Co-encadrants		
Laurent Duval,	Chef de projet,	IFP Energies nouvelles
Frédéric Payan,	Maitre de conférences,	Université Côte d'Azur
Invités		
Christophe Preux,	Chef de projet,	IFP Energies nouvelles

Titre Raffinement en résolution et précision pour la compression et la simulation de maillages volumiques en géosciences

En bref,

La recherche est entrée dans une ère marquée par l'utilisation intensive des données. La quantité croissante d'informations scientifiques pose un défi dans plusieurs domaines d'application. C'est le cas notamment en simulation, où l'utilisation d'énormes volumes de données créent des goulets d'étranglement lors de calculs haute performance. Plusieurs méthodes de réduction des données sont actuellement à l'essai pour tenter de résoudre ce problème.

Dans ce travail, nous nous concentrons sur la modélisation géologique et le worflow de simulation en ingénierie réservoir. En géosciences, les modèles sont composés d'informations hétérogènes, notamment la géométrie du maillage et les propriétés pétrophysiques. Ces derniers peuvent contenir jusqu'à plusieurs millions de cellules. Des techniques d'upscaling/upgridding sont couramment utilisées pour réduire le temps de simulation. Pourtant, elles sont souvent ad hoc et ne répondent pas entièrement à tous les besoins de manipulation des données : visualisation, stockage, et génération d'une donnée à résolution et précision adaptées pour la simulation.

Nous proposons une méthodologie complète permettant un raffinement en résolution et en précision, basée sur HexaShrink, un outil de décomposition multi-échelle basé sur les ondelettes. Nous évaluons ses capacités de compression sans perte puis avec perte en le combinant avec des codeurs entropiques génériques et évolués (type "zerotree") , ainsi que sa pertinence visuelle en appliquant la méthode sur une collection de maillages représentatifs.

Nous testons aussi de manière approfondie l'impact du raffinement de la résolution et de la précision sur la simulation. Tous nos tests de simulation ont été réalisés sur Lundi, un modèle représentatif de différents environnements géologiques, généré spécifiquement pour ce travail. Les résultats se comparent positivement aux codeurs de référence SZ et ZFP, en définissant des métriques d'évaluation objectives corrélées aux évaluations subjectives des résultats de simulation réservoir.

Mots-clefs maillages volumiques, multi-échelle, ondelettes, compression, simulation, ingénierie des réservoirs

Title Refinable resolution and precision for volume mesh compression and simulation in geosciences

In short,

Research has entered a data-intensive era. The growing quantity of scientific information challenges several application domains. That happens notably for the simulation science, where huge data volumes create bottlenecks in high-performance computing workflows. Several data reduction methods are currently being developed as potential answers

We focus on geological modeling and reservoir simulation workflows. Geoscience models are composed of heterogeneous information, including mesh geometry and petrophysical properties, and may contain several millions of cells. Upscaling and upgridding techniques are commonly used for simulation time reduction. Yet there are often ad-hoc, and do not fully answer all the data manipulation needs: visualization, storage and ultimately well production prediction at the appropriate resolution and precision. We propose in this work a comprehensive methodology with refinable resolution and precision based on HexaShrink, a wavelet multiscale decomposition. We assess its performance on size reduction and visual relevance with lossless and lossy compression in a benchmark of meshes using entropy and zerotree coders.

We also extensively test the impact of refinable resolution and precision on simulation. We specifically designed case-studies based on Lundi, a representative model of different geological environments. Results compare positively with state-of-the-art coders SZ and ZFP, by designing objective performance metrics that correlate well to subjection reservoir production assessment. Finally, we designed a complete workflow including a shareable mesh (Lundi, to be released as open data), designed for simulation in reservoir engineering. This model (meant to serve as a benchmark) can be processed by our multiscale decomposition, and then compressed by different encoders (lossless, or progressive/lossy) and compared to alternative compression methods. The resulting compressed meshes, generated at refinable resolution and precision, can be processed, and their quality can be evaluated at several steps of a simulation workflow.

Keywords volume mesh, multiscale, wavelets, compression, simulation, reservoir engineering

Remerciements

Pour ce travail de thèse, j'ai pu bénéficier de l'aide et du soutien de nombreuses personnes appartenant au cadre universitaire, professionnel et personnel.

Je tiens à exprimer mes plus sincères remerciements à mes encadrants. Marc ANTONINI, mon directeur de thèse au laboratoire I3S à Sophia Antipolis ainsi que Frédéric PAYAN dont le soutien dans les moments les plus difficiles m'a permis de poursuivre mes travaux. Je remercie tout particulièrement Laurent DUVAL, mon promoteur de thèse IFPEN, dont le nombre d'heures consacré à ce projet et à mon bien être général a très fortement été revu à la hausse. Vous avez, à n'en pas douter, démontré des compétences dans un encadrement à distance désormais plus qu'utiles. Merci de m'avoir fait découvrir le domaine de la compression et les disciplines associées, j'espère dans ma future carrière pouvoir partager l'essence et les bienfaits de ces traitements.

Un grand merci à Christophe PREUX pour son expertise dans le domaine de l'ingénierie réservoir et à Sébastien MAGAND pour son étude du big data et de son usage en simulation venue consolider la partie contexte de ce projet.

Thanks to my reviewers and examiners for reading and reporting on the manuscript, and for the enriching exchanges during the defense.

C'est avec émotion et beaucoup de regrets, en l'absence de pot de départ en raison de la crise sanitaire, que je quitte toute l'équipe IFPEN de la Direction Sciences et Technologies du Numérique de Solaize. Je tenais à vous remercier pour votre accueil chaleureux et l'atmosphère quotidienne bienveillante, notamment Laurent PIGEON pour sa patience dans la résolution de mes problèmes informatiques. Merci au bureau des doctorants pour leur étude sur la théorisation des humeurs durant ces trois années de thèse partagées et vécues aux côtés de Johan l'ancien, David, le tic de tic et tac, et les jeunes Bassel et Adam dans l'ordre de succession. J'espère tous vous revoir très prochainement.

Je tiens aussi à remercier l'équipe I3S pour avoir rendu mes déplacements à l'Université Côte d'Azur agréables, notamment : Cyprien, Melpo, Arnaud, Danny, Eva...

Pour finir, je remercie ma famille qui a su m'entourer et m'encourager : mes parents, sans aucun doute mes plus grands supporters, mon frère, ma sœur, leurs conjoints respectifs pour l'exemple qu'ils m'ont toujours donné, ma grand-mère, pour sa force de caractère dont j'espère avoir hérité, Philippe et Marie-Do pour m'avoir plus que choyée lors de mes séjours à Antibes, Sylvain pour m'avoir accompagnée sur cette fin de thèse, ma belle-famille pour m'avoir accueillie comme leur fille/sœur notamment lors du premier confinement, mes ami(e)s de géologie et de médecine, du collège/lycée et plus particulièrement ma Marie. Enfin merci à toi mon Gautier, une étape de plus pendant laquelle tu as su me supporter, écoutant le récit quotidien des réussites et des échecs, sachant trouver les mots justes pour me rassurer et rendre le moment plus doux. On a pas fini de grandir ensemble, j'ai hâte de la suite.

Résumé

Les sciences nous aident à appréhender le monde qui nous entoure : qu'il s'agisse du monde vivant, d'objets naturels à petites ou grandes échelles, ou de phénomènes physiques proches ou lointains. Cette connaissance du monde est nourrie par des avancées technologiques et intensivement documentée par une donnée générée en grande quantité. Face à ces volumes de données croissants les capacités d'exploitation deviennent insuffisantes.
Nous nous intéressons dans cette étude à la gestion des données de simulation pour les géosciences grâce à la compression, deux notions qui seront définies et contextualisées par la suite.

Dans cette introduction nous différencierons trois types de données scientifiques courantes : la donnée expérimentale, de simulation, d'apprentissage. La donnée expérimentale est une donnée produite par des outils de mesure dont la précision n'a de cesse d'augmenter (microscope, satellite, méthodes de tomographie). Ils permettent aujourd'hui d'observer et mesurer des objets et des évènements autrefois méconnus ou inaccessibles.

Ces dernières années les progrès du numérique ont démocratisé des outils qui viennent compléter la vision naturaliste fournie par les données expérimentales. Notamment, les outils de simulation approximent des phénomènes physiques dont la structure de base et le modèle mathématique sont connus. Les résultats de la simulation peuvent être visualisés et analysés pour améliorer la compréhension d'un objet : sa formation et son devenir (courants océaniques, phénomènes météorologiques, cosmologiques). Plus récemment les méthodes d'apprentissage ont été popularisées. Ce processus se base sur l'exploitation de bases de données existantes, et ce pour définir des liens profonds et comprendre les rouages de phénomènes complexes.

Qu'importe leur type, le domaine, d'énormes quantités de données ont été générées par des scientifiques en soif de connaissance : planifiant des campagnes d'acquisition colossales, et construisant des centres de calculs aux performances exponentielles. Cette frénésie a été soutenue par la découverte de matériaux innovants (intégrant les terres rares dans leurs compositions), qui ont permis la miniaturisation des outils informatiques, augmenté les moyens, et facilité cette quête du savoir. Les

découvertes et connaissances profondes qui en découlent constituent actuellement le socle et l'avenir de nos sociétés technologiques.

Pourtant certains chercheurs tirent la sonnette d'alarme : les données scientifiques sont générées en volumes exponentiels, des volumes qui ne semblent pas se restreindre aux vues des futurs projets. Or la manipulation de ces grands volumes est problématique à différentes étapes du processus. En effet elle peut altérer des puissances de calcul (simulation, apprentissage), ralentir la transmission de la donnée de son lieux de production/calcul à l'outil d'analyse du scientifique (ordinateur local), sans parler des besoins de stockages croissants sous-jacents. Finalement, ces volumes de données sont visualisés/analysés pour qu'en émanent des résultats parfois difficiles à cibler, qui ont tendance à se noyer dans ces volumes.
Ainsi différentes étapes du processus sont dites pénalisantes et appelées goulets d'étranglement par la communauté scientifique. De plus la gestion des volumes de données entraine un coût significatif de moyens, qu'ils soient matériels, énergétiques et économiques. Ce problème tend à limiter le nombre et l'ampleur des projets en recherche. Pire encore, il contraint actuellement des scientifiques à détruire une partie de la donnée sans même l'avoir analysée.

Pour prévenir et anticiper ce déluge imminent, différentes solutions constituent des axes de recherche majeurs. Ils visent à réduire la quantité de donnée en suivant cinq **con**cepts distincts dont on souligne ici l'imbrication. Premièrement le **com**pressed sensing manipule des bases de données naturelles qu'il échantillonne (spatialement/temporellement ou de manière aléatoire), ceci afin de la résumer par un sous-ensemble. Ce sous-échantillonnage contrairement aux autres méthodes contient *in fine* une donnée intelligible, semblable à la donnée initiale. Les prochaines notions quant à elles se basent sur la transformation algorithmique de la donnée brute. De cette phase émerge des coefficients représentatifs de la donnée initiale. Il s'agit d'une phase dite de **con**centration. S'en suit une **con**densation, cette étape ne conserve que la partie émergente de la donnée transformée pour en extraire moyennes et écarts-types *etc.*. La **com**pression quand à elle intègre la donnée transformée, et génère des fichiers binaires, choisissant ou non de tronquer la précision binaire (quantification). Une dernière étape, dite de **com**paction (encodage), assimile les phases de concentration et de compression, structurant et réorganisant l'information binaire pour réduire le volume de données.

De ces notions imbriquées découle l'intuition qu'une donnée peut être hiérarchisée, et qu'une partie restreinte mais signifiante de la donnée pourrait suffire aux besoins scientifiques. Le but ultime serait alors de pouvoir estimer pour une donnée la quantité minimale d'information utile : à l'image de l'entropie de Shannon - notion fondamentale de la théorie de l'information.
A l'échelle de ce projet de thèse, il serait présomptueux et démesuré de vouloir satisfaire l'ensemble de la communauté scientifique. Cependant nous sommes convaincus que ce travail peut servir différents domaines et inspirer de nouvelles recherches. Ainsi ce projet de thèse s'est concentré sur une donnée de simulation utilisée en géosciences. Nous avons construit une méthodologie adaptée, motivée par des besoins métiers et une connaissance de la donnée, basée sur des notions et outils de compression existants.

Nous présentons dans un premier temps la donnée sur laquelle l'ensemble de notre étude repose. Il s'agit d'un maillage volumique : une structure numérique permettant de modéliser des formations

géologiques profondes. Ce modèle est utilisé pour mieux visualiser et comprendre la mise en place de structures complexes pouvant présenter de très grandes dimensions (bassin sédimentaires, 200-300 km). De plus, cet objet peut servir de support pour simuler les écoulements de fluides au sein d'une formation poreuse (réservoir). De ce fait cette structure est "pleine", c'est à dire composée d'un ensemble continu de volumes unitaires, appelées cellules (hexaèdres dans notre cas). Chaque cellule comporte différents types d'information caractérisant la roche (propriété(s) pétrophysique(s) [valeurs flottantes], âge, nature de la roche [valeurs entières]). Le maillage géologique est donc une donnée hétérogène, puisque constituée de composants différents en terme de nature et de dimensions ($1D$, $2D$, $3D$). L'association de ces données hétérogènes permet une représentation spatiale de l'ensemble des propriétés scientifiques, et à un géologue une analyse éclairée des structures pour proposer une histoire géologique.
En effet, des évènement tectoniques (compression, extension) peuvent engendrer des déformations (plissements) et fracturer la roche soumise à une trop forte contrainte. Ces failles sont des objets essentiels, représentés sur un maillage par le décrochement de cellules le long d'un plan (de faille). Or l'ensemble des failles constitue un objet d'intérêt, puisqu'il devient un chemin préférentiel d'écoulement des fluides, il est donc essentiel d'en tenir compte.

Afin de modéliser des objets de grandes dimensions et d'augmenter le réalisme, les maillages actuelles peuvent comporter jusqu'à des millions de cellules. Cette tendance bien que louable engendre des problèmes de visualisation et de simulation. Il s'agit là de problèmes connus et étudiés dès le commencement des géosciences numériques,dans les années 90. Pour simplifier la manipulation de cette donnée, une approche consiste à réduire la résolution du maillage tout en conservant sa forme et ses caractéristiques géologiques. Appelé *upscaling* (*upgridding*), ce procédé développé en ingénierie réservoir vise à diminuer le nombre de cellules dans le maillage pour réduire le temps de calcul d'une simulation et garantir ses résultats. Du côté de la littérature scientifique, cette méthodologie fait écho au principe de réduction couramment utilisé en simulation (afin de réduire le nombre d'échantillons) dans différents domaines.

Un outil de décomposition multi-échelle, nommé HexaShrink (Peyrot et al., 2019), a servi de base à l'ensemble de notre étude. À partir d'un maillage initial, HexaShrink génère des maillages à plus faibles résolutions. Cette méthode avait pour but premier de faciliter la visualisation de grands maillages, en proposant un affichage progressif multi-échelle, et en préservant pour chaque sous-résolution les discontinuités structurales observées dans la donnée initiale. Cet outil est venu combler un manque de l'état de l'art.
Les résultats d'HexaShrink ont été comparés aux options d'*upscaling*/*upgridding* disponibles dans un *geomodeller* référent. À résolution inférieure, le réseau de failles est gommé et présente des artéfacts, tandis qu'HexaShrink le préserve.

Pour se faire, l'outil associe plusieurs méthodes adaptées à chaque composant du maillage et basées sur la notion d'ondelettes. Cette approche n'augmente pas la quantité d'information, conformément au principe de décomposition en ondelettes. Pour illustrer cette transformation on prend l'exemple d'un signal $1D$ constitué de n échantillons, la décomposition en ondelette génère deux sous-ensembles (ou sous-bandes) chacune composée de $n/2$ coefficients : une partie dite approximation, semblable à la donnée initiale à la résolution inférieure, et la partie complémentaire appelée détail, composée d'échantillons de faibles valeurs. Cette décomposition peut s'opérer de manière récursive,

et dépend du nombre de niveaux de décomposition. L'ensemble de la donnée décomposée représente une structure imbriquée et ordonnée. La conservation des différents niveaux de détail garantit par transformation inverse, la reconstruction d'une donnée identique à la version initiale. Ce concept a été adapté à l'ensemble des composants du maillage volumique. La structure HexaShrink conserve donc les volumes de données initiaux, et présente par conséquent un réel atout dans notre projet.

Un premier travail a consisté à tester la décomposition HexaShrink pour la visualisation puis dans un but de compression sur divers maillages géosciences. Les maillages utilisés se distinguent du fait de leurs dimensions, topologie, certains présentant des discontinuités structurales, mais aussi de leur nombre et type de propriétés attachées. De cette manière nous évaluons les performances de notre transformation HexaShrink sur une donnée scientifique. Pour se faire nous avons calculé l'entropie de la donnée transformée (indicateur de l'efficacité de la transformation) mais aussi utilisé de manière appliquée des encodeurs génériques existants (gzip, bzip2, LZMA), capables de compacter l'information transformée. Ces outils sont dits *génériques* car ils peuvent traiter des données de diverse nature (texte, image, son *etc.*) et ne sont donc pas optimisés pour une donnée scientifique composée de valeurs flottantes de forte dynamique.
Ces codeurs sont "sans perte", c'est à dire qu'ils compressent et décompressent la donnée à l'identique comparé à la version originale. Cette pratique se distingue d'une compression dite *avec pertes*, qui vise à augmenter les performances de compression en supprimant une partie de l'information (par quantification de la donnée transformée). L'altération de la donnée décompressée peut être imperceptible et considérablement améliorer les performances de compression.

Ces différents encodeurs *génériques sans perte* ont assuré les étapes de **com**pression et **com**paction de notre donnée scientifique. Pour vérifier la pertinence de la décomposition HexaShrink nous avons appliqué notre méthodologie à un benchmark composé de sept maillages géosciences.

Quel que soit le maillage, l'utilisation de la décomposition HexaShrink augmente les performances des encodeurs génériques sans perte.
Nous observons cependant que les performances de compression des maillages diffèrent selon l'encodeur. Leurs résultats coïncident intuitivement avec leur année d'apparition. Ainsi l'encodeur le plus récent, LZMA, obtient de meilleurs taux de compression suivi dans l'ordre chronologique inverse par bzip2 et gzip. De plus l'efficacité de la compression dépend de la complexité du modèle traité. Notre maillage le plus simple obtient un taux de compression proche de 12 :1, en utilisant la méthodes HexaShrink/LZMA. Cette même méthode appliquée à un maillage plus complexe (présentant des reliefs, failles) ne parvient qu'à réduire la quantité de donnée par 2. En détaillant les résultats selon les composants, on observe que notre méthode appliquée aux propriétés pétrophysiques n'est pas efficace, au contraire la transformée tend à augmenter la quantité binaire (comparé à l'utilisation simple d'un encodeur).
Cette observation corrobore les résultats de la littérature, traitant de la compression menée sur de la donnée scientifique. Sans perte, les taux de compression n'excèdent pas 2 :1. Une transformation considérée efficace pour la compression d'une donnée multimédia, n'a qu'un impacte limité voir négatif sur une donnée flottante, à grande dynamique.

De ce constat, il nous est paru essentiel d'utiliser une méthode de compression plus adaptée en

choisissant un encodeur évolué. Les encodeurs génériques testés jusqu'à présent n'ont tenu compte ni du volume de la donnée, ni de la décomposition multi-échelle opérée par HexaShrink.
A l'instar de JPEG2k nous avons complété la décomposition en ondelette d'un encodeur progressif type *zerotree*. Cet outil exploite l'organisation multi-échelle héritée de la transformation. Se dessine en effet des structures, qui ont pour origine des coefficients dans l'approximation (ou sous bandes de haut niveau), là où se concentrent les fortes valeurs et s'étendent dans plusieurs sous bandes de bas niveaux vers des coefficients de plus faibles valeurs. Semblables à des arbres, ces structures peuvent être remplies de zeros à la lumière de la profondeur binaire d'où le terme *zerotree*. Des plans de bits les plus significatifs aux plans de bits les moins significatifs, la donnée est progressivement encodée, et les arbres de zéros remplacés par un seul symbole. De cette manière le budget binaire est économisé et la compaction de la donnée optimisée.
Le flux binaire généré par l'algorithme *zerotree* est donc ordonné selon sa *signifiance*. De cette manière la transmission même partielle d'une partie de la donnée encodée permet par décodage et transformation inverse de reconstruire une version approchée de la donnée initiale, à précision numérique inférieure. Le reste de l'information ajoute du détail lors de la reconstruction, son utilisation complète garantit une reconstruction de la donnée à l'identique.
Aux résultats obtenus avec des encodeurs génériques (LZMA, bzip2, gzip) sur des données pétrophysiques [valeurs flottantes], nous comparons ceux générés grâce au *zerotree*. Bien qu'intéressants les taux de compression en sans perte sont limités comme le relaye la communauté scientifique.

Par la suite, nous décidons de changer la précision de nos propriétés pétrophysiques. Nous poursuivons dès lors l'hypothèse initiale qu'une partie de l'information peut être suffisante pour des besoins scientifiques en jouant sur les notions de précision numérique ou résolution spatiale.

En modifiant la précision d'une donnée on modifie sa qualité. Ce changement peut être évalué de manière objective ou subjective. L'évaluation objective se base sur un calcul mathématique, mesurant la différence entre la donnée originale et la donnée dégradée. L'évaluation subjective quant à elle se base sur l'opinion d'un panel d'observateurs non-qualifiés. Selon leur appréciation (visuelle, auditive..), une note subjective est attribuée à la dégradation.
Contrairement à la donnée multimédia, la donnée scientifique peut servir de support à la simulation (notre cas). Par conséquent son altération n'est pas seulement visuelle, mais son impact doit aussi être évalué dans un *workflow* de simulation et estimé par des professionnels. En complément des métriques objectives classiques, nous développons de nouvelles méthodes adaptées afin d'évaluer avec justesse la qualité d'une donnée scientifique. La confrontation des résultats obtenus avec différentes métriques a pour but de valider ou rejeter certaines méthodes d'évaluation, plébiscitées par des travaux sur la compression de données scientifiques.

A propos de la littérature, différents outils de réduction et compression ont déjà été intégrés avec succès à des *workflows* de simulation dans différents domaines (climatologie, cosmologie). La réduction et la compression avec perte permettent de considérablement réduire la quantité d'information binaire, et ainsi soulager différents points clés de la simulation HPC.
L'application de ces méthodes reste encore marginale, car son impact est difficile à évaluer. Cependant, le sujet gagne en popularité au vue de récentes publications, du partage de benchmark et de la diffusion de codes open source. De plus de nombreuses conférences fleurissent à ce sujet.

Pour poursuivre notre étude nous avons créé un *workflow* de simulation (écoulement en milieu poreux) pour évaluer dans un premier temps la méthode multi-échelle HexaShrink, et étudier dans un second temps l'impact d'une précision raffinable.
Nous construisons un cas d'étude comportant un réservoir faillé, baptisé LUNDI, sur lequel nous paramétrons une simulation d'écoulement, classiquement réalisée en ingénierie réservoir pour simuler l'exploitation d'hydrocarbures. Étant amené à répéter un nombre de fois conséquent la même simulation, les dimensions de LUNDI ($128 \times 128 \times 32$ cellules) sont restreintes, pour ne pas être contraint par les temps de simulation excessifs. Aussi les trois failles que comporte LUNDI permettront de tester HEXASHRINK, et d'évaluer ses performances d'*upgridding*. Concernant les propriétés pétrophysiques deux environnements de dépôt ont conditionné la modélisation de quatre jeux distincts, chacun composé de deux propriétés volumiques : porosité - perméabilité. Les deux environnements s'inspirent de la donnée SPE10 (Christie and Blunt, 2001), un modèle réservoir utilisé pour évaluer l'upscaling. L'environnement fluviatile comporte des objets saillants, appelés chenaux. Or l'*upscaling* de cette propriété est périlleux, car la modification des chenaux peut considérablement modifier l'écoulement des fluides dans le réservoir. Les trois autres environnements sont moins contraignants, la distribution spatiale des propriétés y est plus diffuse.

LUNDI est modélisé à l'image d'un *quarter five spot model* : un cas d'école en ingénierie réservoir. La forme générale du maillage est marquée par une pente, dont on peut évaluer le pendage le long d'une de ses diagonales en surface. Pour simuler l'exploitation d'un champ pétrolier, on place au coin le plus bas un puit injecteur opposé à un puit producteur, situé sur le coin de plus haute altitude.
Initialement le reservoir contient deux phases verticalement séquencées : une phase huileuse surplombant une phase aqueuse. L'huile située dans la partie haute du reservoir est extraite par l'injection d'eau dans la partie basse par le puit injecteur. Sous pression, la phase aqueuse pousse l'huile jusqu'au puit producteur où elle est extraite.
Les paramètres de simulation (pression, débit de production) restent fixes pour chaque expérience afin d'assurer une continuité, et pouvoir comparer les résultats des simulations.

Les résultats sont des paramètres physiques mesurés aux puits qui renseignent l'ingénieur réservoir de l'avancement de la production. Dans cette étude, nous nous concentrons sur la courbe “water cut". Elle évalue l'évolution de la saturation en eau des volumes extraits durant l'exploitation du champs. Un remaniement des données d'entrée peut modifier le profil de la courbe. Son analyse sert à valider des méthodes d'*upscaling*. En effet, le changement opéré sur le maillage doit peu ou ne pas impacter les résultats de simulation obtenus avec un maillage haute résolution.
Les différentes versions de LUNDI, les plus basses résolutions générées avec HexaShrink ainsi que la donnée à précision raffinable sont évaluées selon leur courbe “water cut". Dans la littérature, s'agissant des études d'upscaling, les courbes sont tracées pour permettre à un oeil expert de se faire une opinion quant à la méthode utilisée. Ici nous proposons d'automatiser cette phase de validation en définissant des enveloppes d'acceptabilité autour de la courbe “water cut" référence. De l'enveloppe la plus proche “identique" à une enveloppe plus éloignée, nous définissons cinq critères subjectifs pour évaluer les résultats de simulation.

Quatre niveaux de décomposition HEXASHRINK sont appliqués au maillage LUNDI. Lors du pro-

cessus de reconstruction, quatre résolutions intermédiaires sont générées et testées par simulation. Comme détaillé avec la méthodologie HexaShrink, le maillage à la $i^{\text{ème}}$ résolution, noté res.-i est issu de la donnée d'approximation du $i^{\text{ème}}$ niveau de décomposition. Ce maillage contient $(2^3)^i$ fois moins de cellules que la donnée initiale. Nous utilisons les propriétés pétrophysiques de l'environnement diffus, et observons l'impact de l'*upgridding/upscaling* d'HexaShrink sur les résultats de simulation. L'impact pour res.-1 est modéré, l'évaluation subjective donne des résultats acceptables, mais ne le sont plus à partir de la res.-3. Le temps de calcul de simulation diminue considérablement avec le nombre de cellules, passant de 2 heures pour la haute résolution à un calcul quasi instantané pour la res.-3. Ainsi nous démontrons que la décomposition multi-échelle HexaShrink obtient des résultats acceptables et diminue considérablement le temps de calcul de la simulation. D'avantages de tests pourraient conforter son utilisation pour manipuler des grilles peu complexes en ingénierie réservoir.

Dans un second temps nous étudions l'impact de la compression avec perte dans notre *workflow* de simulation. Nous traitons dans cette expérience la haute résolution ($128 \times 128 \times 32$) de la perméabilité de lundi, à laquelle on applique un compandor, avant de procéder à la transformation en ondelettes, et générer la donnée à différents niveaux de précision (zerotree, quantification). L'utilisation d'un compandor est réfléchie et se justifie par des pratiques courantes de modélisation basées sur les lois de la physique.
Pour visualiser nos résultats on diminue la précision numérique de nos données dont on rapporte l'évaluation objective en fonction de la quantité binaire de donnée compressée utilisée. La courbe est tracée pour différentes métriques objectives et complété par l'évaluation subjective des résultats de simulation. Ainsi on vérifie que l'évaluation objective de la donnée concorde avec la qualité des résultats de simulation obtenus avec. Nous concluons sur le fait que certaines métriques, notamment absolues, ne sont pas corrélées aux résultats de la simulation.
Nous démontrons que l'utilisation d'un compandor améliore la qualité objective/subjective et les performances d'un outil de compression. De plus notre approche est comparée à d'autres outils (SZ (Cappello et al., 2019), ZFP (Lindstrom, 2014)), connus et reconnus, optimisés pour la compression de données scientifiques en HPC. Leurs résultats sont comparables à ceux obtenus avec notre approche.

Aussi, nous avons étudié l'anisotropie des données en géologie, ce pour améliorer notre méthode de compression. Les roches décrites par les propriétés pétrophysiques sont en effet façonnées par des évènements naturels anisotropes (sédimentation soumise à des courants orientés, pression lors de l'enfouissement du matériel, autres évènements tectoniques). Le considérant nous avons testé une méthode d'ondelettes en paquet, orientée selon l'anisotropie de la propriété. Cette décomposition particulière nécessite l'adaptation de l'algorithme *zerotree*, étudiée dans ses travaux par Christophe et al. (2008). Cette prise en compte de l'anisotropie améliore davantage la compression de cette propriété.

Dans un dernier temps à l'image de nouveaux outils (TTHRESH (Ballester-Ripoll et al., 2019)) qui étudient les seuillages optimaux des méthodes de compression (SZ, ZFP), nous confrontons l'ensemble de nos résultats objectifs/subjectifs, avec l'évolution de paramètres de compression mesurés

lors de l'encodage zerotree (bit rate per bit plan). Ce qui nous permet de valider expérimentalement notre hypothèse initiale, à savoir : qu'il existe une quantité minimale d'information capable de rendre compte d'une donnée scientifique. Cette intuition fondamentale testée sur un nombre réduit de cas pratiques nécessiterait davantage de tests.

Pour conclure nous avons démontré dans cette étude l'efficacité d'HexaShrink : un outil de décomposition multi-échelle développé pour améliorer la visualisation d'une donnée de grandes dimensions et faciliter son traitement : transmission, stockage. La méthode de décompositon basée ondelettes est complétée d'outils de compression progressive évolués, capables de générer une donnée scientifique à précision réduite. Produite à partir d'une quantité d'information réduite, nous avons montré qu'elle était suffisante pour les besoins de la simulation. Il est envisagé d'en faire usage dans de futurs travaux afin d'optimiser un *worflow* de simulation HPC.
Cette approche appliquée aux géosciences peut être adoptée dans d'autres domaines scientifiques, prenant toujours en compte les besoins métier, et la connaissance de la donnée manipulée.

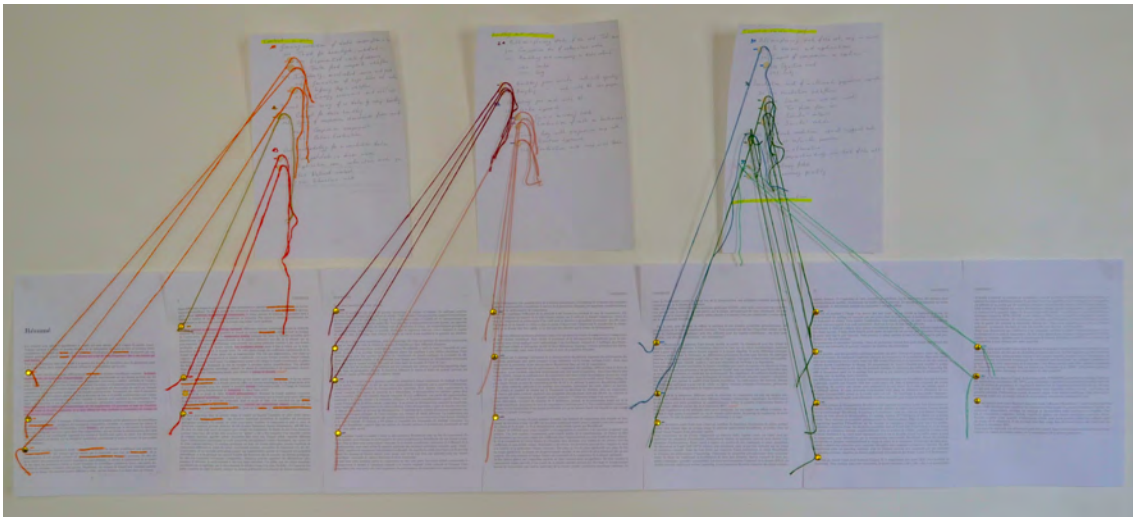


Figure 1: Decorating my living room while writing.

Publications and communications

Journal paper

- Peyrot, J.-L., Duval, L., Payan, F., Bouard, L., Chizat, L., Schneider, S., and Antonini, M. HexaShrink, *an exact scalable framework for hexahedral meshes with attributes and discontinuities: multiresolution rendering and storage of geoscience models.* Computat. Geosci. (May. 2019).

In writing

- Bouard, L., Duval, L., Payan, F., Preux, C., and Antonini, M. *Accurate reservoir simulation with refinable precision on petrophysical properties.* Computer & Geosci. (to be submitted, Feb. 2021).

International conferences

- 81st EAGE Conference and Exhibition (European Association of Geoscientists & Engineers), London. (2019). Bouard, L., Duval, L., Preux, C., Payan, F., and Antonini, M.
  Lossless progressive compression of meshes for upscaling and upgridding in reservoir simulation with HexaShrink. Accepted for an oral presentation.

National conferences

- CORESA (COmpression et REprésentation des Signaux Audiovisuels), Poitiers. (2018). Bouard, L., Duval, L., Payan, F., and Antonini, M.
  Decomposition multi-échelles de maillages 3D hexaédriques dans le domaine des géosciences. Étude des performances en compression sans pertes.

- RING meeting (Research for Integrative Numerical Geology), Nancy. (2019) Bouard, L., Duval, L., Preux, C., Payan, F., and Antonini, M.
  A lossless to lossy compression of meshes for upscaling and upgridding in reservoir simulation with HexaShrink.

- CORESA (COmpression et REprésentation des Signaux Audiovisuels), Sophia Antipolis. (2021, postponed). Bouard, L., Duval, L., Payan, F., and Antonini, M.
  Étude comparative de l'impact d'un codage à précision raffinable sur des données de simulation en géosciences.

Other communications

- LIRIS, Lyon. (June, 2020), Bouard, L., Duval, L., Payan, F., Preux, C., and Antonini, M.
  Comparative study of the impact of refinable-precision coding on geoscience simulation data.
- IFP Energies nouvelles, Rueil-Malmaison. (December, 2018), Bouard, L., Duval, L., Payan, F., Preux, C., and Antonini, M.
  Progressive Compression of Large Volumetric Meshes: Geometry & Properties.

Contents

Acronyms

Meshes & components	
VM	volume mesh
GM	geologic mesh
GVM	geologic volume mesh
RM	reservoir mesh
CAD	computer-aided design
P	generic continuous property
Objective evaluation metrics	
$\mathcal{L}r$	relative euclidean distance
MRE	mean relative error
maxRE	maximum relative error
$\mathcal{L}$	euclidean distance
MAE	mean absolute error
maxAE	maximum absolute error
MSE	mean square error
RMSE	root mean square error
nRMSE	normalized root mean square error
SNR	signal noise ratio
Λ-SNR	compandor-signal noise ratio
PSNR	peak signal noise ratio
SSIM	structural similarity index measure
Binary writing	
MSB	most significant bit
LSB	least significant bit

Compression vocabulary

CR	compression ratio
LOD	levels of details
MRA	multiresolution analysis

Preprocessing

Λ	compandor
V	expandor

Tools

DWT	discrete wavelet transform
CDF 9/7	Cohen–Daubechies–Feauveau
ZT coder	zerotree coder

Simulation

HPC	high performance computing
CFD	computational fluid dynamic
WC curve	water cut curve

CHAPTER 1

General introduction

"Data is the new oil"

Clive Humby, 2006

Technological advances have raised our knowledge and structured the scientific process into a workflow. Its different steps are fed by data, notably numerical data used for simulation, the topic of this study. However, whatever the stage of the workflow, the data quantity increases exponentially. This is demonstrated by the endless deployment of data centers and the use of ever-increasing computing resources. Today's scientific projects process a thousand times larger data quantities than fifteen years ago. The energetic, economic and ecological impact of their use encourages scientists to improve data processing by reducing data quantity. Several methods exist, in particular compression, widely developed for processing multimedia data. Some of these methods improve the gain by adjusting the precision of the data. This change is evaluated by quality metrics.

Researchers seek to understand the surrounding world using technological tools. Innovations from measuring instruments and intensive development of computational resources allow us to better apprehend our environment. Knowledge in diverse scientific fields leaped forward: from materials sciences to cosmology, for climate analysis or geosciences. Objects and phenomena at variable dimensions, difficult to study in the past, are now accessible through digital technologies. Prior to their development, sciences were documented by empirical descriptions and theoretical models. Numerical tools has considerably broadened the field of possibilities by first enriching measurement tools with digital components creating ever more sensitive sensors. Then, the rise of computational resources made it possible to simulate complex natural phenomena such as multi-phase flows, used to study large objects as hurricanes, the flow of liquid phases in the subsoil or on smaller scale air flow around a sail.

This constant improvement of tools and methods considerably increased the volume of experimental data. The growing number of exascale[1] projects in various scientific fields demonstrates our insatiable thirst of knowledge. While progress and discoveries seem infinite, they are limited by concrete aspects as handling of huge data quantities. From public side, the actual cost of what is called “data" is largely ignored and even growing awareness among scientists, high precision in quest for realism is the baseline from modeling to simulation, and leads to exponential rise of data volume and computing resources. The increase in data may give hope for an increase in the volume of knowledge, however, it could drown out the information.

Regarding simulation, the topic of our study, beyond difficulties in processing huge data quantities, it is foreseen that calculation supports (as supercomputers) are reaching some limits. While computational resources are improving rapidly for years, storage and bandwidth capacities are actually more limited. It consequently limits several steps of the simulation workflow and slows down the overall computation performances. Later in this section, we contextualize simulation within numerical environment and communicate the real costs to highlight the emergency of the situation.

To provide partial answers to the problem of data quantity in science, we focus on compression, popularized in the 80's in the multimedia context. We adapted and evaluated methods to propose a data representation with tunable precision for needs in reservoir simulation.

1.1 Data feeds scientific workflow

To address scientific issues, huge data quantities are daily generated and analyzed. Among them, there are different types with distinct structure and binary format. Whatever the type, we find them at distinct steps of scientific workflows. Their increasing consumption is illustrated by the worldwide development of ever more data centers.

[1] Adjective refers to the computing resources, among 10^{18} floating operations per second delivered by supercomputers.

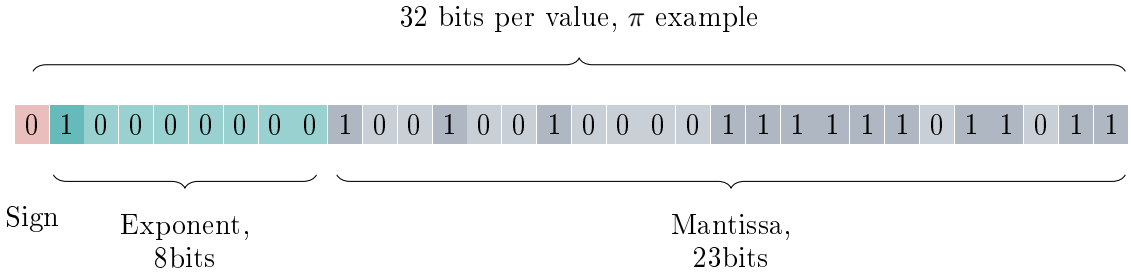


Figure 1.1: π number represented with IEEE-754 floating-point standard, equal to 3.1415927 based on single precision.

1.1.1 Data types

For scientific data, the widely used standard/norm is IEEE-754, used for calculations at floating-point precision. Behind this standard is the number of bits required to write each value. For simple floating-point precision, a value is coded on 32 bits, as illustrated with π example on Figure 1.1. Among the 32 bits, one bit is dedicated to the sign while eight others are used for exponent and 23 remaining correspond to the mantissa. They allow to write a number with seven digits after the decimal point. Other extended standards exist, they multiply the binary size by two or four. They are respectively called double or quadruple precision. This use increases the dynamic range and bring respectively a decimal precision of 16 and 34 digits. It is notably used in simulation for ever more accurate calculation needs, but is currently questioned by scientists. Because for certain applications a fixed precision may not be required and moreover would increase data volumes. For several years concurrent formats of IEEE-754, as POSIX, adjust the length of mantissa to change the precision of the data. However habits die hard, because change in standard would involve to redesign the data chain from visualisation to storage such as the data workflow.

Throughout classical workflow, three types of data can naturally be considered: experimental data, simulated data and “learning data", distinguishable because of their production. Experimental data is the result of a measure or observation made on real objects or phenomena. Considering geological field, seismic and well data are precious and unique because respectively recovered during huge and expensive acquisition campaigns and drilling operations. By contrast simulated and learning data can be generated on request by code execution but are nonetheless expensive as we will see later. Simulated data yields realistic representation of natural phenomena as flow motion description. While learning data is composed by data about real events and used to train a machine for decision-making tasks, or generated by training during the process learning example. Three data types are highly linked and feed entire workflow. Experimental data are used in a first step for modeling and then inform simulation, while learning processes can enrich and correct the original model.

From the three types of data we can use in a workflow, we focus on simulation and differentiate the inputs, the pending data (used for calculation) and the outputs. For simplification, we interest first in simulation inputs. The objective is evaluate the impact of compression on simulation by

using adapted metrics, or in other words to study the simulation integrity at refinable precision.

Returning to a more general point of view, given the quantities of data, it is impossible to ignore the digital infrastructure and overall data management procedures.

1.1.2 Digital infrastructures for data handling

Digital data is handled in the physical world, stored on the disk, loaded into memory, transmitted via optical fiber and analyzed by using CPU-GPU resources. The hardware layer is of great importance in studying the needs for data processing. The most representative infrastructures doing the job are the data centers, whose number has increased considerably in recent years. According to data-centermap.com (updated in 2020), there are around 4500 data centers worldwide in 122 countries, including more than 1800 in the United States. As for France, there are 156 data centers, 35 % of which are in the Ile-de-France (Paris) region.

A data center hosts a set of numerical infrastructures for storage, transfer and computing of data. Combined to supercomputers one can perform High Performance Computing (HPC) to manage massive data quantities and simulate objects or complex phenomena. Handling and processing are now only evaluated in terms of floating-point operations per second (flops) only. For instance, the most powerful supercomputer named Fugaku (Japan) is able to realize more than 400 billiard of floating-point operations per second (400 petaflops). To a lesser extent, the supercomputer used for simulation during our study has 440 teraflops capacity. It is located at IFP Energies nouvelles - Solaize and photographed on Figure 1.2. The website top500.org references performances of worldwide largest supercomputers used for exascale projects[2].
The concern comes from the increasing number of data centers growing in size and resources with each new generation. Consumed power and its impact is becoming a first order matter.

1.2 Infobesity: source variety & growing production

The objective is to extract scientific insight from computed results in the large HPC facilities. According to the famous statement of mathematician Richard Hamming: “The purpose of computing is insight, not numbers”. All these resources and their increase leads to fears of infobesity. By definition, it consists in overload data at the risk losing the data essence and information.

Moreover having previously spoken about flops, we must be aware that the entire power is not exclusively dedicated to the pure mathematical power, but distributed between various handling steps. First of all the memory context is transferred from the HPC through the network, to be saved in a reliable storage (Naksinehaboon et al., 2008), waiting for its analysis in a next step. The required time is mainly underestimated, and the storing cost as well. Obvious solution for speeding up the computing process is to increase supercomputer resources and flop capacities. Nevertheless such approach saturates global digital infrastructure and reveals system limitations and bottlenecks.

[2]https://www.exascaleproject.org/ registers and solutions american exascale projects.

Figure 1.2: On left, a picture of the IFP Energies nouvelles supercomputer, called ENER440, at Lyon - Solaize, with details of its back (right picture) and its cooling doors equipped with cold water circuit 13 degrees.

Additionally in simulation because the data can be regenerated on demand, the cost it represents and the notion of value disappears. The codes are therefore executed daily and data is produced massively, with great precision in a permanent quest for realism. Intermediate results, often voluminous, raise storage issues. Therefore some HPC simulation run are merely destroyed, because their retrieval is not feasible. This represents both an energetic waste, and a loss in simulation history and analysis.

It is crucial nowadays to consider the full data processing and report the actual price caused by the infobesity. Indeed, it seems inconceivable within the current trend to ignore the ecological cost of the so-called digitalization and its handling dependencies their energetic consumption (for cooling computers) and their CO_2 emissions for the user-cost.

Our concerns are the more or less hidden cost of all these operations. Let us also briefly mention, from a material point of view, the tensions around the rare earths necessary to produce ever powerful computers and the around infrastructures. It contributes to the deterioration of the already complex geopolitical situations in producer countries. Cost of digital is therefore a growing concern for energy, economic as well as ecological and political reasons.

1.2.1 Scientific projects with oversized data

Here are two examples of experimental projects collecting huge data volumes analyzed by Big Data, from the micro to the macroscopic scale. We talk about genomic investigations and go next to the

sky to focus on stars and objects of the cosmos.

In genomics, technological advances have reduced the cost of DNA sequencing. From an estimated cost of 100 M$ in 2000 for the sequencing the entire genome of a human, this cost decreased to 1000$ starting in 2016. This lower cost has made it possible to study more and more cases. As example, project of United Kingdom's Biobank plans "to sequence the genomes of 500 000 volunteers and follow them for decade" to give a more complete picture of society-wide health according to Pavlichin et al. (2018). However, the data generated by the sequencing of an human genome represents "ten to hundreds of gigabytes of data". Such figures demonstrate why this field promises to be one of the most expensive in terms of storage.

Changing of scale: Digital Sky Survey (DSS) illustrates the scientific data consumption considering the astronomy. It is a huge atlas of sky pictures "started in 2000, [which] collected more data through its telescope in its first week than had been amassed in history of astronomy" (according to *the Economist*). Obviously, quantity of data for scientific usage did not increase progressively, but exponentially. Considering the current world's largest digital sky survey (PAN-STARS, 2019), it collected 1.60 PB of data during the four first years, which represents "30 000 times the total text content of Wikipedia"[3].

Far from being reassuring, the largest simulations currently generate up to several petabytes of data in a single run, as the example of the code developed by NCAR [National Center for Atmospheric Research] for analysis of climatology presented in Subsection 4.1.1. The size of its outputs (CMIP [Coupled Model Intercomparison projects] current version of Phase6) has been multiplied by one million in the space of 20 years from 0.50 TB to hundreds of petabytes in 2020.

This is not only question of data size, but these large volumes result in multiple sub-problems throughout the workflow.

1.2.2 Bottleneck & issues in data workflow

In pushing the process and level of detail for simulation, scientists have pointed out some limits for data handling. While these issues were known and investigated for data visualization, storage and transmission, other steps have recently been identified as bottlenecks to HPC performance.

As example to illustrate past transmission difficulties, we can cite a simulation code, named POP [Parallel Ocean Program] (Smith and Gent, 2002), developed by Los Almos National Laboratory and run on supercomputers. Its results were transmitted and analyzed by searchers in local on their desktops. In 2011, for a bandwidth capacity of $1\,\mathrm{MB\,s^{-1}}$, the transfer of a data sets of size between 56 GB and 5.60 GB (daily production) could catch from 12 hours to 50 days, according to the report of Woodring et al. (2011).

Although the bandwidth capacity has been steadily increased (reaching $2.50\,\mathrm{TB\,s^{-1}}$ for IBM Summit, the world's second largest supercomputer), it did not improve enough compared to the

[3] https://hubblesite.org/contents/news-releases/2019/news-2019-12.html

storage/memory size of supercomputers equipment. The latter has indeed increased tenfold over the last ten years, while bandwidth has only improved its capacity by a factor of 2.5. Therefore, as shown by Cappello et al. (2019), the HPC performance of the largest supercomputers is currently limited by data transfer during simulation, which creates bottlenecks (as for checkpointing/restart approach explained in Subsection 4.1.2). We could also mention the need for in situ visualisation (mainly studied), code execution acceleration *etc.* improved using recent compression tools later discussed in our literature review Subsection 4.1.2.

These bottlenecks mentioned in HPC studies are proofs of the limitation of a digital ecosystem facing the rise of data production. Their deleterious effect is also perceptible at a more global level by studying energetic, economic and ecological indices.

1.2.3 Energetic, economic and ecological cost

Let's start with some general figures successively considering BigData, HPC and data centers. Current forecast predicts the field of BigData analytic will represent a world market of 90 billion dollars [4] in 2025, and experiences a very strong growth of 20 % in the next five future years [5]. In regard HPC market, it is estimated to 60 billions dollars in 2025 [6] with a growth of 7 % [7]. [8]

According to recent figures on global energy consumption, digital has been estimated at 1.9 % in 2013 and 3.3 % in 2020 with a forecast growth of 9 % per year (Ahvar et al., 2019). Only considering data centers, facilities have huge energy requirement as cooling post representing among half of their needs. It represents 1 % of electrical energy consumption in 2005, 1.8 % in 2012 and potentially will reach 5 % in 2030. This alarming observation is fueled by new technologies as machine learning (García-Martín et al., 2019). In the flood anticipation research is increasingly focusing on the benefits of compression for saving space in data centers, and therefore energy (Raïs et al., 2019).

Germany is actively working on this ecological and economical problematic (Geveler et al., 2015) in view of energetic transition of the country. Various studies in particular led by DKRK (Deutsches Klimarechenzentrum - German Climate Computing Center) (Hübbe et al., 2013; Kunkel et al., 2014; Kuhn et al., 2016) seek comparing various compression tools, and economical return by storage optimization. Kunkel et al. (2014) estimated at this time at 144 000€ per year the price of storing 5.60 PB of data dedicated to earth Science system. Among processing methods classically studied to save space, he concluded compression is the most promising for the laboratory needs.

[4] https://www.statista.com/statistics/254266/global-big-data-market-forecast/

[5] https://www.businesswire.com/news/home/20190627005451/en/Global-Big-Data-Market-Witness-CAGR-19.7

[6] https://www.grandviewresearch.com/press-release/global-high-performance-computing-hpc-market

[7] https://www.marketsandmarkets.com/Market-Reports/Quantum-High-Performance-Computing-Market-631.html

[8] Now all these market growth predictions are subject to change upwards in the light of recent events (majority of references edited in 2019, pre-Covid19).

1.3 Numerical diet for (scientific) data?

Given the growing increase in data production in HPC, scientists address the question by various research axes. They notably explore approaches based on compression, deduplication, indexation (Kraska et al., 2018) and clustering (Geist and Reed, 2016). Advances in those domains are carried out by prominent research centers, for instance in the USA (Lawrence Livermore National Lab., Oak Ridge National Lab., Argonne National Lab. National Center for Atmospheric Research (NCAR)), in China or Japan (Sakai et al., 2013; Kolomenskiy et al., 2018). Large HPC facilities are being developed, notably with exascale computing, able to perform billiars flops. In particular, it is being promoted at the International Conference for High Performance Computing, Networking, Storage, and Analysis, on November 2020. The processing of their outputs has given rise to numerous studies. More recently, the emergence of bottlenecks during simulation, mentioned in Subsection 1.2.2, has led to new questions to overcome them. Scientists seek methods that can be integrated into the scientific workflow (in situ technologies) by satisfying divers applications that comprise it.

So far we pointed several bottlenecks along the simulation workflow considering problematic data volumes, but situation is worst than we believe. The field suffers from a lack of standards, especially in experimental and simulation sciences. Therefore a universal solution is not practicable. Conversely multimedia data (pictures, videos, sound) have classical formats, leading to the development of compression standards that have invaded our daily lives. From there we identify a classical scheme of compression tools, and common methods to evaluate their efficiency.

1.3.1 Legacy of multimedia standards

Compression facilitates the dissemination of information by reducing data size. It is worth noting that multimedia compression supported the deployment of the web. Interestingly, the most used standards for $1D$ signals (audio, known as MP3) and $2D$ images (pictures, with JPEG) were already developed by the beginning of the 1990's. In audio and pictures, individual digital files are relatively small. Standards have evolved more steadily for $3D$ multimedia data instantiated by video (space and time). While storage capacity and network transmission have witnessed an impressive increase over the last decade, data quantity (including duplication) grows exponentially, and disk transfer rates tend to plateau. For instance, a Full HD (high-definition) two-hour movie (1080 lines, 1920 pixels per line, 50 images per second) needs a transmission capacity of $2.49\,\mathrm{Gbit\,s^{-1}}$ for streaming without compression. The entire video file requires $2.49 \times 3600 \times 2$, or 17.91 Tbit for storage! There is no need to mention that video streaming would be impossible without compression. Two types of compression are traditionally distinguished: with perfect information preservation, or allowing some precision loss.

Lossless compression Original data is perfectly reconstructed after compression/decompression. It includes all the archiving techniques, generic such as the zip, and others, dedicated to data types. It is used to store the data and recover it without any loss (documents, codes *etc.*). Compression gains are generally limited to a factor of 2–3.

Lossy compression This approach is widespread in multimedia (audio, pictures, video) because

useful compression rates are mostly higher (8–20 fold in audio or image) and thus better adapted to the usage (*cf.* the above example with HD video). Lossy compression implies a variable data loss, which means the data can not be perfectly reconstructed, and the decompressed data will not be exactly similar to the initial data. Lossy compression involves a control of the data loss with respect to the compression ratio. Using the imperfectness of human perception (vision or audition in the multimedia context), data modifications are allowed, at locations or in frequencies where they are not perceptible, or harmless for interpretation. The MP3 format, for instance, degrades frequencies barely audible to the human ear, or sounds masked by others. Concerning images, edges and textures are especially preserved, because the human eye is highly sensitive to their degradation. Hence, compression gains are much higher, while respecting (hopefully sufficient) visual or auditive quality.

1.3.2 Data compression components

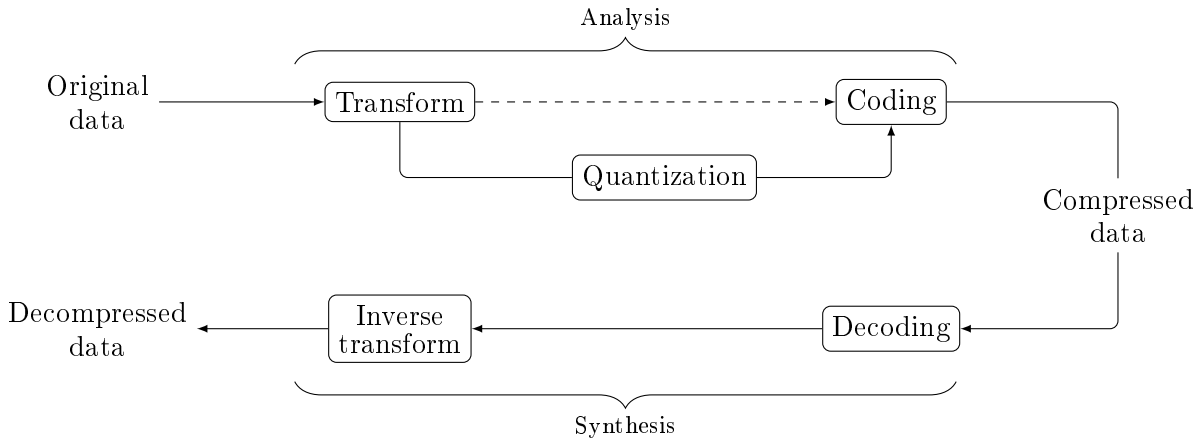


Figure 1.3: Classical compression/decompression pipeline.

Most of the time, compression and decompression follow several main steps, named components in this study and illustrated in Figure 1.3. Firstly, the input data is generally *transformed* (or predicted). The idea is to project original data onto a new space, better suited to compression. An efficient transform will decorrelate the data by changing values with coefficients close or equal to zero, because sparse data would be easier to handle and store. Transform efficiency could be measured by parameters as entropy (Shannon)[9] : it assesses the improvement of data representation after transformation. There are different transformations, among the most widely used we can mention discrete cosine transform (DCT), discrete Fourier transform (DFT), discrete wavelet (DWT) transform. The latest transform has been popularized in compression for multimedia by JPEG2000 (Taubman and Marcellin, 2002).

[9]Mathematical estimation of the minimal binary quantity required to transmit a sample of an information.

The second step is the *quantization*, that converts the transformed data into a finite number of *symbols*. By simplification of small coefficients, sparsity of the data is further increased to reduce binary size of data and facilitate its handling. This use is for lossy compression only (MP3 example). It significantly improves compression performance while modifying precision of decompressed data.

Then, a *coding* step exploits the remaining redundancy in the quantized data, to further compact it into a binary file. It can be performed with quantities of entropy coders, the most popular being developed by Huffman (1952). We could also cite *dictionary* approach, whose the best representatives belong to Lempel-Ziv family (LZMA for example, used in Subsection 3.4.1). Roughly, all these methods look for replacement of repeated symbols by shorter one to compact the information, and finally to save bit-budget.

To better compress data acquired at various sampling rates, and to allow efficient decompressions at different resolutions (think about screen sizes for seamless video viewing), recent standards have investigated data transformation using the concept of multiresolution. It permits to represent the data at different embedded (lower) resolutions (corresponding to "low frequencies" or *approximations*), complemented by several *detail* components (or "high frequencies"). Details and approximations are combined to reconstruct higher resolutions. Wavelet transforms are instances of such techniques, as applied in our study (*cf.* Section 3.2).

As said, multimedia already possess many compression standards, well-known to the general public: MP3 for music (developed in the MPEG-1/MPEG-2), JPEG for images, based on a discrete cosine transform (Wallace, 1992), and all the MPEG-derived standards for video: H264, HEVC for the more recent ones (Ohm and Sullivan, 2013). While some of these standards are "old", possibly less efficient than more recent techniques, they remain *de facto* standard, because they are widespread. For instance, JPEG2000 (Taubman and Marcellin, 2002) previously mentioned outperforms JPEG in quality, through its wavelet-based implementation, but did not find its expected success in mainstream multimedia. In other words, maximum quality is not the only driver for acceptance. Usability is also an important criterion.

1.3.3 Classical performances evaluation

Beyond usability, scientist should be picky about other measurable criteria to compare methods and evaluate their performance. In this section, main metrics are listed according to the type of compression.

Lossless compression is assessed in term of execution speed and compression ratio (CR). Last parameter is related to the bits rate, which consists in the bit number used to code a sample of compressed data. Depending on the user's needs, one of the parameters may be more important. If the need is storage, emphasis is given to compression performance because as previously saw in Subsection 1.2.3, space is precious and expensive. But when considering transfer and visualization, execution times become the main concern. For these applications compression and decompression should have a minimal impact on overall time. Some methods, even though based on same notions

such as Lempel Ziv (dynamic dictionary) tools, obtain very different performances, due to the optimization of particular components. For example, LZAM is very competitive in term of compression performance compared to LZ4. But LZ4 is the faster of the two (Szorc, 2017).

Lossy compression removes information during quantization step to improve compression performance. It obviously degrades data obtained after decompression. This introduces a dilemma: *the better the compression the heavier the alteration due to the compression* (as illustrated on Lena, Figure 1.4, using jpeg compression standard). While data quantity must be minimal, a certain quality remains appreciable. By consequence, it appears that quality assessment is crucial with lossy compression, and choice of metrics could be more or less adapted to evaluate alteration and to guide suitable compression.

original image
262 Kbytes

10% of quality
7 Kbytes

5% of quality
4 Kbytes

Figure 1.4: Lena (512x512 grayscale image) generated by linux batch conversion, from original (on left) to *jpeg* versions, by keeping 10% (middle) and 5% (right) of the quality, and reducing size of image from 163 Kbytes to 4 Kbytes

There are two ways to assess the quality of decompressed data. First one is objective and based on mathematical principles, while the second is subjective and focuses on opinion of a panel of observers. Their role is to provide marks according to the degree of alteration, called mean opinion score (MOS). Degradation would not be necessary perceptible according to visible and audible spectrum. Such approach is heavy to set up, as human organization, contrary to objective evaluation, as computing approach.

Objective evaluation is easily computerized. It quantifies difference between original and degraded data. A majority of metrics arranged on Figure 1.5, derivate from Euclidean distance defined by $\mathcal{L}_2$, as normalized root mean square error (nRMSE) and variation of signal noise ratio (PSNR, SNR, Λ-SNR). Considering gray scale images for example (whose pixels value ranges between 0 and 255), these metrics operate pixel by pixel, computing difference between original and degraded version. Such an evaluation is valuable and reliable for visual appreciation.

But we express certain reservations about its use for scientific data without prior testing. As depicted by Figure 1.5, we differentiate relative from absolute approaches. Unlike the absolute error, the relative error is related to the original values. Hence, same error made on a sample would be minor if concerns a high value, and major if the original sample is small. As will be seen later in our experimental case (*cf.* Subsection 4.2.1), distinction is necessary and essential. Others metrics could also be tested to evaluate the degradation of the data structure (SSIM [*Structural Similarity Index Measure*], Fidelity, Q criteria).

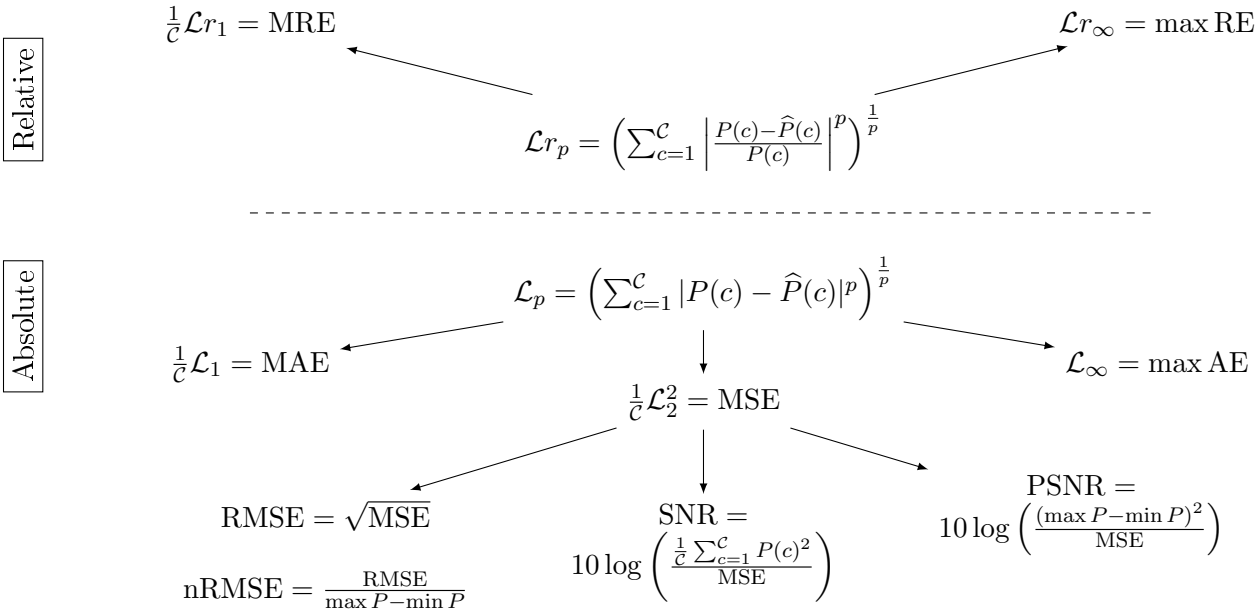


Figure 1.5: Classical objective quality metrics organized around the norms in their relative form $\mathcal{L}r_p$ or absolute form $\mathcal{L}_p$.

Finally, ultimate purpose would consist in correlating the subjective assessment with the objective scale. This in order to guarantee objective thresholds above which the decompressed data will be subjectively acceptable and guide the suitable compression.

In conclusion to this first chapter, a general context has been established by addressing current issues about data production in scientific fields. The data volumes generated in mass are measured in binary quantity and results from precision-resolution often over estimated. Injected at different steps of the scientific workflow, we distinguish experimental from simulation and learning data. Reflect of its over consumption, the increasing number of data centers reflects excess of modern time. To highlight its consequences, we refer to its economic, energetic and ecological cost. In parallel high technologies reveal limits in data management. Compression appears as a promising solution to reduce data quantity. Commonly used with multimedia data, and assimilated by default to standard formats invading our everyday life, the compression pipeline comprises different components whose nature influences its efficiency. Performance is notably evaluated by various parameters. Finally,

compression could be conservative or at refinable precision to increase its gain. In the latter case, it is required to assess the quality of decompressed data in order to evaluate the compression impact.

In this work, compression inspired us to manage scientific simulation data. We restricted the scope to geosciences. Natural context and interest objects, called reservoir, are introduced in the following chapter. Such areas modeled by meshes to execute flow simulation are intensively studied. In geosciences, the meshes are complex composite data hardly manageable, and time-consuming for simulation because of potential large dimensions.

CHAPTER 2

Reservoir modeling context

We focus on the field of geosciences and study deep geological layers. Of great interest to the oil and gas industry, they may contain rocks called reservoirs, whose porosity concentrates energy resources. The profitability of their exploitation notably relies on simulations on numerical models. They consist in meshes integrating structural discontinuities and filled with petrophysical properties. The latter are numerical data characterized by particular dynamics that, sometimes, can be preprocessed with compandor to increase the quality of the compressed data. Our knowledge of the data and of adjacent methods allows us to define the basis of an innovative representation that preserves the geological features, and thus to limit the impact of compression on the simulation.

There are many scientific fields using large data volumes for dedicated applications. Among them, earth sciences are undeniably intensive users, to study large earth objects and stand therefore for domains of choice in our study.

Geosciences are split into sub disciplines because Earth is composed by several envelopes and each discipline will focus on a particular one. From the deeper to the higher, we distinguish the Earth's core, mantle, crust, biosphere-hydrosphere and atmosphere, they are studied by diverse communities from the geology to the climatology. Although resorting to dissimilar objects and models, they may use comparable methodologies. Understanding of physical phenomena occurring there bases on local field measures feeding a simulation model made of equations system. However geosciences could refer to all these natural objects, we only focus on what happen on Earth's crust, and will use for simplification the term "geosciences" for this limited area.

The part of geosciences that is interested in energy production, notably the oil & gas industry, remains of strong economical interest. Hence it keeps on investigating part of the subsoil and improving elements of the workflow used by geologists, geophysicists and reservoir engineers, illustrated in Figure 2.1.

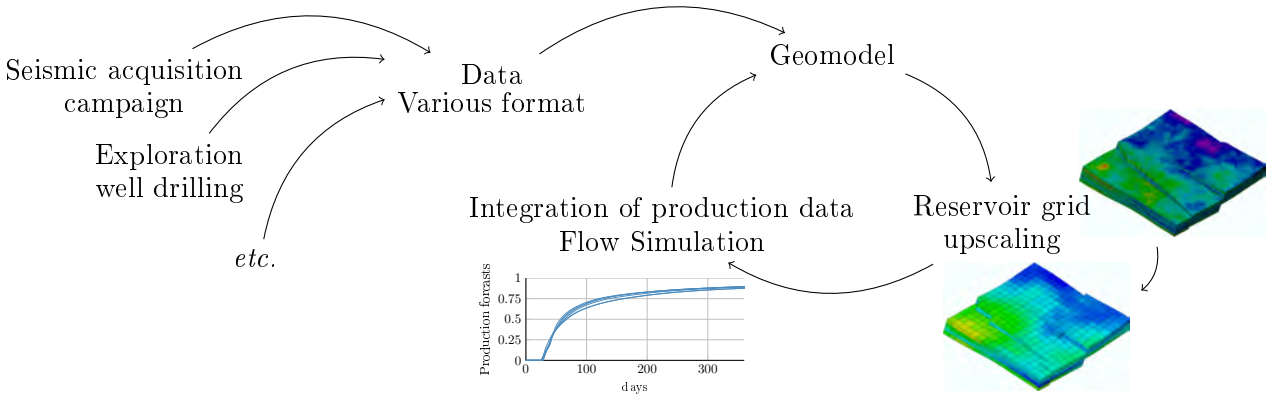


Figure 2.1: Illustration of composite geoscientific workflow. First step (left) is the geologists work. To create a geomodel, they use experimental data measured on the field with different methods. These tools provide data at different format and incertitude levels: from hard data extracted by exploration wells to topographic images recovered during seismic acquisition campaigns. (Right) generated geomodel is not directly usable for simulation and requires the generation of an additional grid at lower resolutions to reduce computation time. Upscaling is operated by reservoir engineers. Then, by simulation are predicted at certain incertitude level the production parameters. Finally, the real parameters recovered during exploitation will enrich and correct the model in a feedback loop. *Terms of the steps of the workflow will be further explained. Although specific to the geosciences, we estimate a generic scientific workflow would be close to this one.*

Throughout this workflow, a complex geological object can be modeled by a geological volume mesh (GVM), mix between a volume mesh (VM) and geological mesh (GM). It corresponds in heterogeneous data composed by diverse components to represent the $3D$ structure and the media

properties. A reservoir mesh (RM) models a particular geological area (introduced in Subsection 2.1.1). RM could be used as support to simulate the flow of liquid or/and gas that happen within. In a quest for realism, it can be extremely detailed and dimensions can be extremely large[1]. If so, their direct simulation would be less tractable because it would be too time consuming. To prevent it, solutions exist (namely upscaling, *cf.* Subsection 3.1.1) in the field to decrease RM dimensions but finally present drawbacks in data management by creating tailored data versions but an additional amount of data. Our study aims at potentially addressing entire workflow by using an original data representation with tunable precision inspired by compression to overcome time and resources constraints. We will use domains based knowledge on GMs and models to properly address each components of meshes. Although integrated and tested for a precise application our methodology is generic and has been designed to be reused by other fields and applications.

2.1 From reservoir formation to numerical model

Our study focuses on sedimentary rocks localized now at kilometers deep. They come from sediments deposed on surface millions of years ago. Elements were progressively collected in particular structures called basins at variable quantity and velocity. In Subsection 2.1.1, we will see that the deposit environment and the nature of sediments influence the structure and petrophysical properties of the future rocks. Note that all information on sedimentary basins is of great interest for oil & gas industry, because these structures can include large area of porous rock, called reservoirs, draining fossil resources (oil and gas) over the years. Profitability of their extraction is determined with help of numerical models. Basins and reservoirs of interest are represented by GMs, introduced in Subsection 2.1.2. They are hypothetical models based on partial field data, integrated to the workflow illustrated in Figure 2.1. Modeling step is crucial, because geological features as structural discontinuities (faults network) may strongly impact the liquids flow and so the simulation. Therefore it seems essential to identify them and preserve them throughout the workflow.

2.1.1 A primer in geosciences for reservoir

The Earth crust could be represented by succession of horizontal rock layers. This stacking structure could be formed in diverse geological contexts. Among them we focus on conditions that give rise to the formation of fossil resources. This geological story started 20 to 350 millions years ago, with progressive deposit of sediments in particular structures called sedimentary basins.

Nowadays exploited, these structures could extend on huge surfaces, up to several hundred kilometers across, and the reservoir technical attainable set between 600 meters to 8 kilometers deep.

They originally consist of depressions in the earth's crust (most of the time) covered by water (oceans, sees, fluvial systems or lakes), where the transport of sediments of variable origin stops. The majority of sediments are detrital and come from continental erosion, but can also be organic or linked to activity of living organism (of animal as plankton shell or vegetal origin). The nature

[1] up to billion of cells

of sediments determines the types of rocks resulting from the sedimentation process.

Each formation (in sedimentary basin and others) could be described according to its deposit age and its lithologic composition, while its physical state would be characterized by petrophysical properties: the rock density, its porosity, its permeability and its phase saturation (water, gas, oil present in the porous system). Such information represents properties data, part of GM components, detailed next in Subsection 2.1.2. Among existing properties, we subsequently focus on two of them: porosity and permeability.

Porosity, noted ϕ, is the percentage of the empty volume V_e on the total rock volume V_{tot}, as described by Equation 2.1,

$$\phi = \frac{V_e}{V_{tot}}. \tag{2.1}$$

While permeability, noted K, evaluates the ability of rock to be crossed by liquids under a gradient of pressure noted $\vec{\nabla}P$. The Equation 2.2 is an extension of Darcy law for multi-phase flow. It computes for a liquid phase (water, oil, gas) its velocity, $\vec{v}$, in function of its relative permeability, kr, its density, ρ and its viscosity, μ, subject to a permeable media and the gravity, $\vec{g}$. K is measured in Darcy (homogeneous to one surface, one Darcy is equal to $9.87 \times 10^{-13}\,\mathrm{m}^2$).

$$\vec{v}_= - \frac{kr}{\mu} K \left(\vec{\nabla}P - \rho\vec{g} \right). \tag{2.2}$$

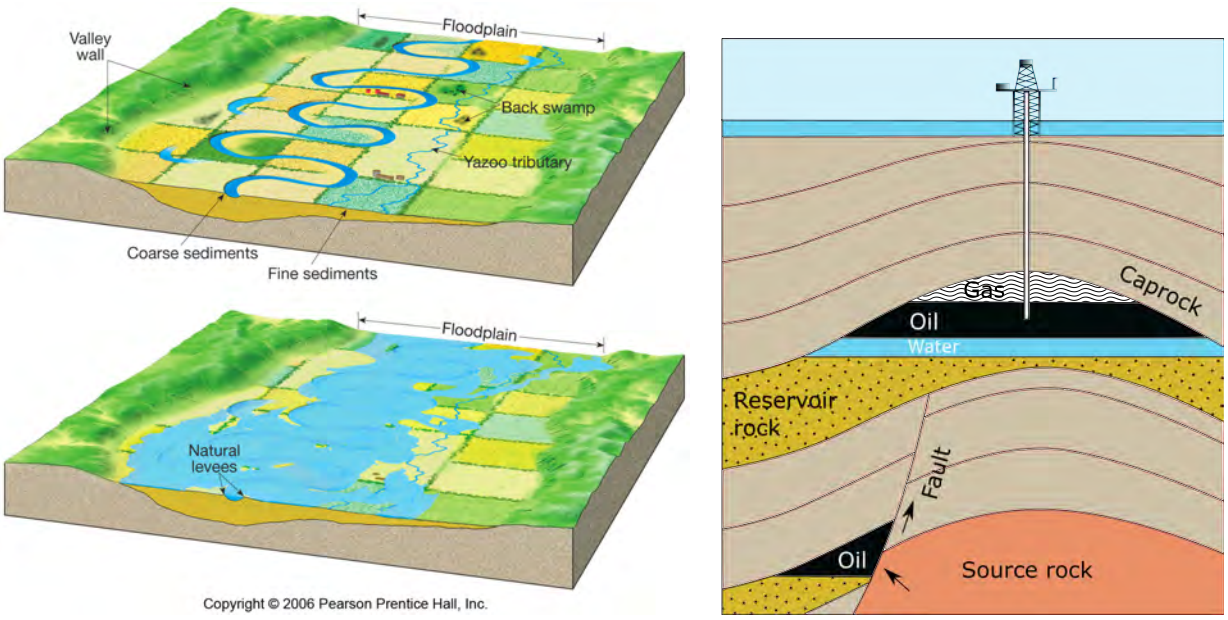


Figure 2.2: Illustration of geological structures, with fluvial deposit environment (left) and schematic geological cross section of basin, and resources localization (right).

These physical quantities have very different numerical orders of magnitude. As the histograms

of porosity and permeability will next illustrate in Figure 3.20. These parameters are governed by physical laws (as power law, or exponential relation *etc.*), the regularity of which is confirmed during laboratory tests. As example, the basic $K-\phi$ law in reservoir geosciences is effective in many cases. It regulates relation between porosity and permeability and consists in Equation 2.3 with A and B criteria experimentally assessed.

$$K = A \exp(\phi) + B. \tag{2.3}$$

In addition, certain ranges of values have a strong impact on flow simulation. As small ones, whose space accumulation can create impermeable structures. Their placement is inherited from the time of sediment deposition.

Deposit environments example Depending on the deposit environment, very different structures could be initiated, as illustrated with fluvial example on left Figure 2.2. Difference between two pictures of left-figure is the river water level, leading to distinct deposits at the same place. In the upper left-figure, low water flow digs a channel and distributes the coarse and light sediments according to the velocity felt in different parts of the channel. In case of flood illustrated in bottom left Figure, water exits the riverbed and floods the neighboring plain, called flood plain. A fine deposit of sediments will form there the future silt and alluvium formations. These rocks have low porosity & permeability and constitute natural impermeable border along channels. Made of coarser elements, channels system is composed by sand formations with high porosity & permeability. It therefore forms a preferential flow way in the subsoil in case of fluvial system.

Maturation process, generation and storage of hydrocharbon Coming back to the formation of hydrocarbon resources, we know that the raw material is rich in carbon. It is mostly issued from decomposition of living matter deposed on the basin floor. To be preserved from oxygen and decarbonisation, very specific conditions need to be set up around carbon matter. This requires fast covering of organic elements by impermeable layers of mud, creating anaerobic environment. This progressively allows to bury high quantity of carbon under successive sediment formations.

At depth, under extreme pressures and temperatures, organic sediments are changed into source rock, as shown on the right of the Figure 2.2. Then starts the formation of oil and gas during cooking process. These hydrocarbons will next migrate along faults across covered formation up to porous layer called reservoir rock, whose porosity is naturally occupied by water, but replaced by oil and gas during migration. In particular stratigraphic conditions[2] if impermeable formation (named caprock) covers reservoir formation, hydrocarbons stop their progression and accumulate in this structure called trap.

Across geological time, formation features and structures are susceptible to evolve. Already complex with regard to the petrophysical structures (channels) initiated during sedimentary deposition, horizontal layers can be modified by tectonic events (compression, extension) creating folds and

[2]submit of anticline structure: type of fold that is an arch-like as illustrated on right Figure 2.2.

faults. As saw with hydrocharbon migration, these objects considerably influence the liquid phase flow. The understanding of the field allows us to pay attention to these details. Although such small values appear insignificant relative to the global mesh scale, it is crucial to represent for GM modeling and processing.

Considering software environments developed to manage GMs (such as Openflow, Gocad, Petrel) we notice their interest with regard to physical scales, and illustrate it by two following examples. First, from a numerical point of view, the permeability property is commonly visualized with an exponential color scale instead of linear to focus on small values. Secondly, from structural point of view, it is possible to exaggerate the vertical scale to inflate the thin geological layers to make them accessible.
Then in next section, is detailed a mesh structure and its components, commonly used by geologist to represent complex geomodel and tend to preserve geological features.

2.1.2 VM for reservoir modeling

VMs discretize the interior structure of $3D$ objects, sedimentary basin and reservoir in this study. They partition their inner space with a set of three-dimensional elements named cells. While pyramid and triangular prism partitions exist, most of the existing VMs are composed of tetrahedral (4 faces) or *hexahedral* (6 faces) elements. They are called tets or hexes (sometimes bricks), respectively. A VM composed of different kinds of cells, tetrahedra and hexahedra for instance, is termed hybrid. A VM is described by the location of vertices in $3D$ space (*geometry*) and the incidence information between cells, edges, and vertices (*connectivity*). In function of the application domain, VMs also contain *physical properties* (petrophysical properties in geosciences) associated to vertices, edges, or cells.

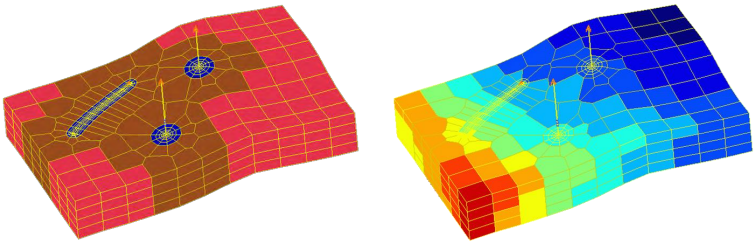


Figure 2.3: Example of a GM used (left); same mesh with the associated porosity property (right). Note that this mesh is hybrid and unstructured, with both hexahedral and tetrahedral elements.

A non-degenerate hexahedra has 6 *faces* named quads, 12 *edges*, and 8 *vertices*. Depending on incidence information between cells, edges and vertices, hexahedral meshes are either unstructured or structured. The degree of an edge is the number of adjacent faces. An hexahedral mesh is *unstructured* if cells are placed irregularly in the volume domain, *i.e.*, if degrees are not the same for all edges of the same nature. Unstructured meshes have an important memory footprint, as all the connectivity information must be described explicitly. However, they are well-suited to model complex volumes, Computer-aided design (CAD) models for instance, as shown in Figure 2.4.

An hexahedral mesh is *structured* if cells are regularly organized in the volume domain, *i.e.*, if

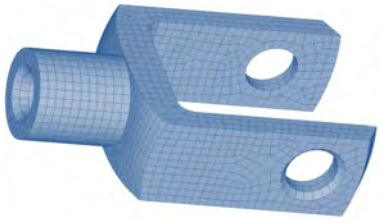


Figure 2.4: CAD model defined by an unstructured VM.

the degree is equal to four for *interior* edges (inside the volume), and equal to two for *border* edges (on a border of the volume). In that case, the set of hexahedral cells is topologically aligned on a $3D$ Cartesian grid (see Figure 2.5). Each vertex of the mesh can be associated to a node of the grid. Hence, each cell can be indexed by only one triplet (i, j, k), and the connectivity information becomes implicit: only the position of the vertices is needed to model the mesh.

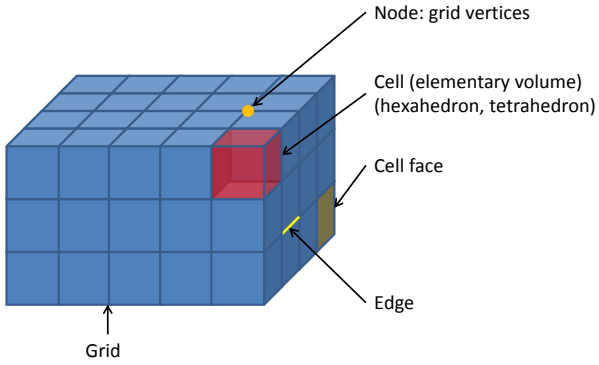
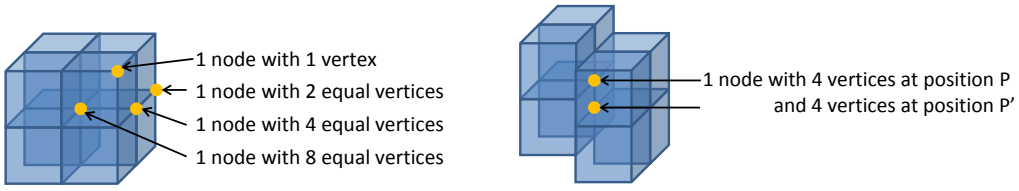


Figure 2.5: Structured hexahedral mesh composed of $(5 \times 4 \times 3)$ cells.

RMs used to perform simulations could include structural discontinuities and property particularities. Such features require specific handling, especially for a multiscale representation (*cf.* Subsection 3.1.1).

Geometrical discontinuities In geosciences, hexahedral meshes are generally structured, and thus based on a Cartesian grid. But these meshes may contain *geometrical discontinuities*, corresponding for instance to geological faults. It induces a vertex disparity in space at the same node. The association of one node of the Cartesian grid with 8 vertices (one for each adjacent cell) handles this specificity. Figure 2.6-(a) provides an illustration of a fault-free volume. On Figure 2.6-(b), we see that this structure allows to describe for instance a vertical fault (by positioning vertices differently about the node), while preserving the Cartesian grid.

The most popular data structure for structured hexahedral meshes with geometrical discontinuities is the *Corner Point Grid* tessellation of an Euclidean $3D$ volume: a structure developed by Ponting (1989) and still currently used (Røe and Hauge, 2016; Lie, 2016). This structure is often termed *pillar grid*. It is based on a set of vertical or inclined pillars running from the top to the bottom of the geological model. A cell is defined by its 8 adjacent vertices (2 on each adjacent pillar,



(a) Free-fault area. (b) Area with a vertical fault.

Figure 2.6: A fault-free and a fault area.

see Figure 2.7), and the vertices of the adjacent cells are described independently, in order to model faults and gaps. Across the associated Cartesian grid, each cell can be indexed by a triplet (i, j, k).

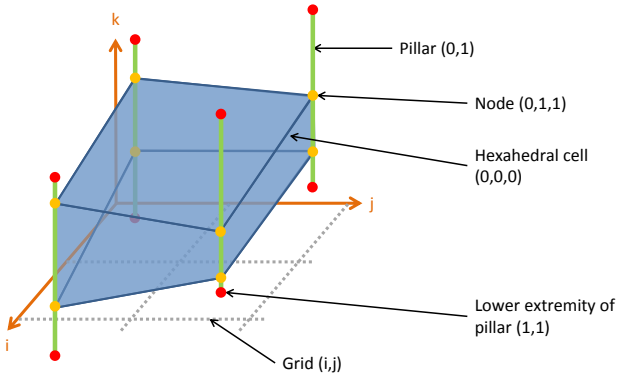


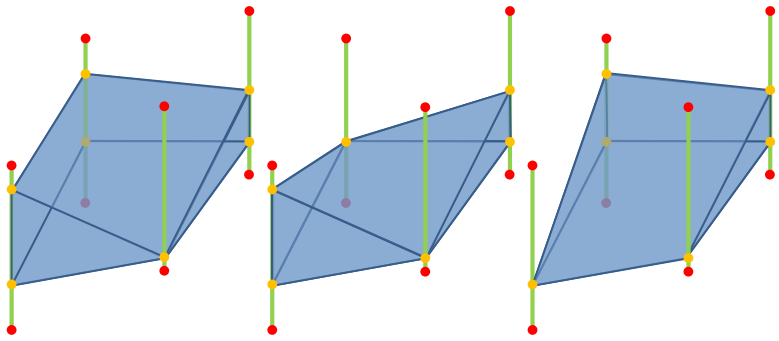
Figure 2.7: An hexahedron, according to the *pillar grid* structure.

This pillar grid also allows to model geological collapses (or erosion surfaces), by using *degenerate* cells, i.e., cells with (at least) two vertices on one pillar located at the same position (see Figure 2.8 for different degenerate configurations).

Globally, the description of the structure represents a large part of overall mesh sizes, between one or two thirds for the GMs we studied. The remaining data consists in mesh properties attached to the cells, describing their activity or various petrophysical properties: continuous or categorical.

Mesh properties Firstly, the *cell activity* is a Boolean parameter notably used to deactivate cells and model irregular borders, or used to focus on an interest zone.

Secondly, there are *categorical* ($\mathbb{N}$) and *continuous* ($\mathbb{R}$) properties. Such properties are spatially distributed in the mesh: each cell is linked with k discrete values and n floating-point values corresponding respectively to the k categorical properties and the n continuous properties. Their distribution is determined during mesh modeling by geostatistical tools (variogram, kriging) and based on field measurements (typically well and the seismic data). The categorical properties describe for instance the geological formation repartition (rock nature, deposit age). The discrete value is more a label which could corresponds to various parameters, not rationally quantifiable or manipulable. The continuous properties describe petrophysical properties attached to the cell, such as the porosity and the permeability, defined in Subsection 2.1.1. Such parameters have a huge amplitude range



(a) Single. (b) Two opposed. (c) Two adjacent.

Figure 2.8: Degenerate hexahedral cells due to a single collapsed pillar (a) or two different collapsed pillar locations (b), (c).

and require to preserve a high numerical precision. Considering for comparison an other quantity as the values of pixels of an image. For black and white version, a gray scale is commonly used and featured by a certain binary level, this means a certain number of bits, noted $nbits$, will be used for each pixel. For a gray scale at 256 binary levels, a pixel intensity value necessitates thus one byte to be coded ($\log_2 256 \Leftrightarrow 8\,\text{bit}$); contrary to a physical value, whose numerical accuracy uses 7 digits and requires on average up to 24 bit.

2.1.3 Using dynamic of reservoir property

As we plan to work on compression, we have to take into account the binary representation. We know that on practice the data is quantified due to the limited precision of computers. Given this limitation, we have chosen to exploit it by giving the advantage of precision to certain ranges and saving bits budget on others. As an example, we consider a large area with an almost constant permeability of 1000 mD, if a value were to increase by 2 mD, it would have no impact on the flow. After a certain level of permeability, the material allows the liquid to pass through. Conversely small values will have a strong impact on the flow rate, if we add 2 mD to an initial value equal to 10^{-5} mD, this could create an artifact and have an impact on the flow. This knowledge can be used to our advantage, instead of an evaluation based on absolute error, we will focus on relative scale. The logarithmic transform will allow to linearize a power law and to preserve a relative deviation.

Referring to the literature, it is known that this approach has already been used for the processing of audio data. It allows coding fewer bits because the transform makes the data more homogeneous. This treatment, called compandor (Li et al., 2005), noted Λ, is used to preserve whispering in a loud sound environment. Compandor echoes to expander, noted V, the inverse method to return to the original distribution.

We propose in this study an original version of the compandor noted $\Lambda_{\alpha,nbits}$, parametrized by α and $nbits$, expressed by Equations 2.4, with the original property denoted P.

α can vary between 0 and 1. Equal to one, $\Lambda$is a simple linear function (stay in absolute scale), but tends to logarithm functions with α approaching zero (moving to relative scale).

The factor $2^{nbits} - 1$ permits to convert the real values to discrete values distributed between 0 and $(2^{nbits} - 1)$. Thus, the choice of *nbits* allows us to clearly set the data precision.

$$\Lambda_{\alpha,nbits}(P) = round\left(\frac{\lambda_\alpha(P) - \lambda_\alpha(\min P)}{\lambda_\alpha(\max P) - \lambda_\alpha(\min P)} \times (2^{nbits} - 1)\right), \tag{2.4}$$

with

$$\lambda_\alpha(P) = \begin{cases} \log(P+1) & \text{if } \alpha = 0 \\ \frac{(P+1)^\alpha - 1}{\alpha} & \text{if } \alpha > 0 \end{cases}$$

Whereas the expander $\mathrm{V}_{\alpha,nbits}$ is expressed as follows:

$$\curlyvee_\alpha(P) = \begin{cases} \exp(P) - 1 & \text{if } \alpha = 0 \\ (\alpha P + 1)^{\frac{1}{\alpha}} - 1 & \text{if } \alpha > 0 \end{cases}$$

$$\widehat{P} = \mathrm{V}_{\alpha,nbits}(\Lambda_{\alpha,nbits}(P)) = \curlyvee_\alpha(\frac{\Lambda_{\alpha,nbits}(P)}{(2^{nbits} - 1)} \times (\lambda_\alpha(\max P) - \lambda_\alpha(\min P)) + \lambda_\alpha(\min P)). \tag{2.5}$$

To demonstrate the compandor effect, we use a vector composed of eight values between 0 and 20 000 displayed on the first line of the table 2.1. The data is a representative sample of the permeability property later introduced on the Figure 3.20, partly composed of small values lower than hundred and sharing the same upper limit. By applying the $\Lambda_{\alpha,nbits}$, we illustrate the interest of using an adapted preprocess to preserve precision of the lowest values at a comparable *nbits*. We start with 2 bits and incrementally increase *nbits* up to 4. Binary writing of a value is vertically scaled from the most significant bits (MSB) to the least significant bit (LSB). The *nbits* increment adds an LSB layer and increases the accuracy of the results by using the $\mathrm{V}_{\alpha,nbits}$ to return to the original distribution.

We test on the first column a linear function by using α equal to one, and compare results applying logarithm function by using α equal to zero on the second column. Focusing on the original value 10, we observe that for a linear preprocess, the result is always equal to 0, whatever the *nbits* used. In comparison, by applying V_0 and by increasing the *nbits*, result for 10 is refined from 26.10 to 15.90 to 13.00. Even considering 10 000, half of the maximal value, result from $\Lambda_{0,4}$ use is closer to the original than the result of using $\Lambda_{1,4}$, respectively equal to 10 334.10 and 10 666.70.

Then, we consider the Λ_0 helps to better preserve the precision of the values of the sample, as regards the values of the lower half, by passing from a linear scale to a relative scale. What is an opportunity in our scientific context as in other fields dealing with similar data[3].

By revisiting the necessary precision to obtain realistic simulations, we decide to propose an adaptive representation combining the *numerical* with *spatial* (multiscale) precision.

[3] Although it does not seem obvious what data would need this type of treatment. Objectively, we could start with the data visualized using logarithmic scales.

Original data		[0 0.1 1 10 100 1000 10 000 20 000]	
		$\alpha = 1$	$\alpha = 0$
$\Lambda_{\alpha,2}$		[0 0 0 0 0 0 2 3]	[0 0 0 1 1 2 3 3]
MSB	1	0 0 0 0 0 0 1 1	0 0 0 0 0 1 1 1
LSB	0	0 0 0 0 0 0 0 1	0 0 0 1 1 0 1 1
$V_{\alpha,2}$		[0 0 0 0 0 0 13 333.3 20 000]	[0 0 0 26.1 26.1 735.8 20 000 20 000]
$\Lambda_{\alpha,3}$		[0 0 0 0 0 0 4 7]	[0 0 0 2 3 5 7 7]
MSB	2	0 0 0 0 0 0 1 1	0 0 0 0 0 1 1 1
	1	0 0 0 0 0 0 0 1	0 0 0 1 1 0 1 1
LSB	0	0 0 0 0 0 0 0 1	0 0 0 0 1 1 1 1
$V_{\alpha,3}$		[0 0 0 0 0 0 1142.9 20 000]	[0 0 0 15.9 68.7 1179.8 20 000 20 000]
$\Lambda_{\alpha,4}$		[0 0 0 0 0 1 8 15]	[0 0 1 4 7 10 14 15]
MSB	3	0 0 0 0 0 0 1 1	0 0 0 0 0 1 1 1
	2	0 0 0 0 0 0 0 1	0 0 0 1 1 0 1 1
	1	0 0 0 0 0 0 0 1	0 0 0 0 1 1 1 1
LSB	0	0 0 0 0 0 0 0 1	0 0 1 0 1 0 0 1
$V_{\alpha,4}$		[0 0 0 0 0 1333.3 10 666.7 20 000]	[0 0 0.9 13.0 100.7 735.8 10 334.1 20 000]

Table 2.1: $\Lambda_{\alpha,nbits}$ effect on $1D$ data stream, changing α value (column), as the bits number (row).

2.2 From storage to simulation

Problematic Beyond the difficulties of processing (storage, transmission, visualization) large data volumes, some GMs could serve as inputs for simulation (RMs). The process gathers diverse heterogeneous components describing structure to petrophysical properties. Each component is featured by specific dimension and dynamic range. Their combination models a geological object used to simulate the flow of liquid phases (oil, gas, water) occurring inside. However, the simulation is not directly executable on the detailed mesh, but imposes an additional step in workflow by creating a supplementary data at lower resolution to run a faster simulation. Several *ad-hoc* methodologies already exist and reduce RM dimensions by upgridding/upscaling the grid and the attached properties, as detailed later in Subsection 3.1.1. However, this approach does not appear to be the most efficient to data processing as it multiplies the data volumes and pipes in the workflow. As regards the multimedia, compression tools based on multiresolution approaches are promising options, but are rarely used for simulations.

Proposition From these observations, we propose an approach inspired by tools from various fields and our geoscience knowledge that could integrate the workflow. We are thus working on an multiscale representation for GVMs dealing with components heterogeneity. For this purpose we propose a method adapted to the data dimensions and dynamic range. In addition, we plan to devote a larger share of the binary budget to a particular range, the accuracy of which is required for simulation, this by changing notably the absolute scale into relative.

The change in scale and respect of the physical laws governing our properties influence our evaluation metrics developed to ensure the preservation of data at refinable precision (spatial & numerical). Using adapted metrics we evaluate the method for different applications, from storage to visualization and simulation.

Figure 2.9 teasers our proposition for GVM representation. Essence of data is progressively condensed across decompositions into smaller red subpart on figure. It facilitates access to the lower resolutions and order the binary data to get the appropriated numerical precision for simulation. Thus, we want to know if huge GVMs can be wisely represented to deal with different aspects taking into account the context.

Plan Our work is divided into two chapters relating to our main contributions. First, in Chapter 3, we introduce a suitable representation for our GVMs, i.e. capable of arranging the data while reducing its binary quantity for compression purpose. Secondly, in Chapter 4, the representation is integrated into a simulation workflow to evaluate the impact of the RM generated at refinable precision (still considering spatial & numerical precision).

To begin, we review in Science the existing representations used to manipulate volume data, starting with geosciences and then broadening the spectrum (*cf.* Section 3.1). By combining the strengths of all methods, we define a dream scalable representation for our complex GVMs (*cf.* Subsection 3.1.4). To lay the foundations of our representation we use HexaShrink, a multiscale decomposition tool for GVMs, preliminary developed for visual purpose, introduced in Section 3.2. We first apply HexaShrink on a benchmark composed by eight GVMs (*cf.* Section 3.3). Their lower resolutions are visually compared to the results obtained with a popular geomodeller. Then to

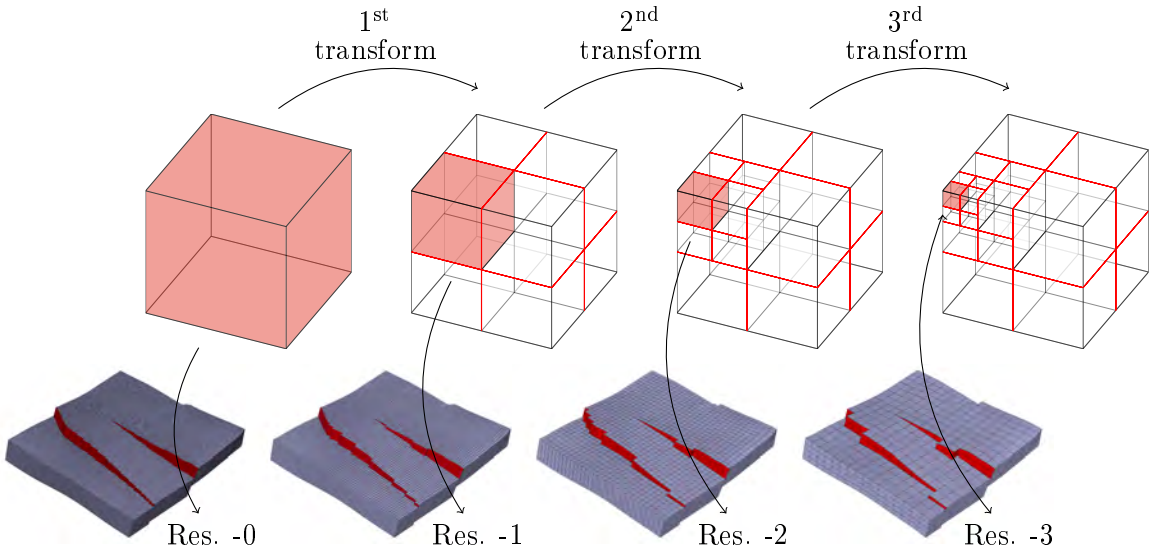


Figure 2.9: Teaser for data volume processing. Transformations are successively applied, to condense information essence (approximation) in red subpart, whose lower resolutions are built from Res.-0 to Res.-3.

evaluate compression performances of the scalable representation we associate HexaShrink with generics encoders in a conservative approach (*cf.* Section 3.4) and finally with an evolved progressive coder, named zerotree, to evolve toward refinable approach (*cf.* Section 3.5). The quest for the suitable refined precision is our objective to improve compression but not only. In Chapter 4, we also assess the precision within simulation workflow.
This second contribution starts with a state of the art on refinable compression in simulation (*cf.* Section 4.1). In this section, we detail some of the tools that will be later used for a comparative study. In Subsection 4.2.1 we model a complete simulation workflow to evaluate our representation first in an upscaling purpose (*cf.* Subsection 4.2.2). Secondly the refinable precision is applied on mesh properties preprocessed or not by compandor to increase the precision on specific ranges (*cf.* Subsection 4.2.3). To validate this processing, innovative metrics are suggested based on the field knowledge and expectations. By combination with standard objective metrics we verify their coherency. We thus seek to identify the parameters to evaluate the precision of data suitable for simulation. We exploit on a last part some data features to improve the process (*cf.* Section 4.3).

CHAPTER 3

HexaShrink, a multiscale representation for geological volume meshes. Application to multiresolution rendering and storage

"Compression... What else?"

Laurent Duval, 2017

Scientific community is interested in various methods for handling large volume data. Regarding the specific field of geosciences, we first mention the upscaling/upgridding techniques, daily used in simulation for "simplifying" geological volume meshes. Then, we make an overview of generic tools developed for compressing volume meshes. Finally, we approach the most popular techniques existing for compressing scientific data in general. By gathering the bright sides of each method, we present HexaShrink, a scalable representation dedicated for the geological volume containing attributes and discontinuities. We show that HexaShrink provides nice renderings at different levels of detail, but can be also integrated in different lossless-to lossy compression workflows, to facilitate the storage of geological volume meshes.

3.1 A variety of representations for volume scientific data

In the first place, this section presents three ways that we explored to put forward a suitable representation for the GMs, and particularly the RMs. Subsection 3.1.1 gives an overview of the upscaling/upgridding methods, well established in geosciences for years in the context of simulation. In a nutshell, it consists in generating a coarse mesh from a given highly resoluted mesh, and to use it to drive simulation. By decreasing the number of cells, simulation time is ineluctably reduced. Second, Subsection 3.1.2 overviews the multitude of compression methods dealing with VMs. The main purpose of these methods is a compact storage, in scientific fields generally not concerned by simulation. Third, Subsection 3.1.3 is devoted to newer compression methods developed for scientific data with floating-point accuracy specifically. Finally, Subsection 3.1.4 summarizes the advantages and the limitations of these three approaches in our context. We also discuss the required features of an "ideal" representation for GMs.

3.1.1 Upscaling & upgridding techniques

For decades now, geosciences have been overwhelmed by data quantity, notably in reservoir simulation. Considering this growing issue, reservoir engineers promoted a research axis to generate coarser grids more appropriate for simulation. The so-called upgridding/upscaling techniques remain current practice in geosciences.

The integration of field measures (well and seismic data) with modern geostatistical tool (Jensen et al., 2000) indeed generates highly resolved computational grids, in which petrophysical properties (porosity, permeability, ...) are spatially distributed. Generally composed of hundreds of thousands of cells, dimensions depend on the required level of detail. Numerical simulations performed on those grids are used to describe the multiphase flow to optimize the hydrocarbon recovery (Coats et al., 1967; Peaceman, 1977; Thomas, 1981) during field exploitation. Simulations should be performed several times to estimate the uncertainty level of the parametrized model (Durlofsky and Chen, 2012), but this could not be performed on the initial mesh.

To reduce the computation times, a coarse grid is generated from the initial data, by upscaling the properties and upgridding the RM. These scale changes involve a merge of connected cells and their attached continuous properties, reducing significantly the number of cells. The porosity is often upscaled by a simple bulk volume-weighted arithmetic average, while permeability upscaling is difficult, because the property is not additive. A generic approach consists in using arithmetic and harmonic averages for the horizontal and vertical permeability, respectively. This method is called static, performing arithmetic operations on property data. It could be sufficient for simpler RMs. Li and Beckner (2000) study the spatial distribution of permeability, properties, and facies to determine the optimal merging, and preserve details on the way of the main flow. However other dynamic methods, based on the flow study are recommended for more complex meshes. To name one (Durlofsky et al., 2000) uses a finer resolution in high fluid velocity area, identifying it by the resolution of local well-driven flow problems. A large number of upscaling methods exists, provided in detail by reviews (Qi and Zhang, 2009). The upgridding or grid coarsening is related to the upscaling result, adapting the mesh structure by merging the cells whose the properties have been upscaled. Quality of upscaling and upgridding can be assessed according to simulation results, as

latter explained in Subsection 4.2.1, or by other metrics taken on upscaled data (Preux, 2014). The main challenge is to choose the more efficient for a given model, the one that will preserve result accuracy, while allowing a significant reduction of the cell number and consequently of computing time.

A wavelet-based upgridding/upscaling technique has been proposed by Mehrabi and Sahimi (1997). Its efficiency has been investigated in many works, and notably on RMs (Rasaei and Sahimi, 2008). The basic concept consists in performing a wavelet transformation on the permeability property, producing blocks of details. Independently, their values are compared to a pre-defined threshold to determine which block is considered as significant during the fluid flow. If a block is considered as non significant, the cells constituting this block are merged. Otherwise, the cells are preserved, to conserve the full resolution in the regions of interest for simulation. Later, Babaei (2013) employed wavelets to provide a coarsening operator during simulation. Hence, the simulations are driven by adapted grids, which reduces the computation times, while preserving satisfactory simulation results. A more recent article (Rezapour et al., 2019) generalizes the method to irregular grids by using graphs, as later explained in Subsection 3.1.3 with the work of Iverson et al. (2012). The irregular grids in geosciences are well known, and represent an elegant way to model reservoir complexities, notably the faults. Nevertheless, the cell indexing is not possible, so the resolution of linearized equation systems for flow simulation are much more complicated.

Discussion These wavelet-based methods do not save the details removed during analysis to get RMs of lower resolution, which could permit a perfect reconstruction by reversing the process (i.e., by synthesis). This pushes the users to store the initial data in parallel.

During the review made on geosciences and the simulation fields, we also note the lack of details on the wavelets used. This contrasts with the current practices in multimedia, where the precise nature and the characteristics of the wavelets are given. We thus consider that, depending on the nature and the regularity of the different elements of our grids, a more rigorous choice of multiscale representations can be beneficial.

In a more global way, the upscaling techniques essentially serve the interest of the simulation providing an additional lighter RM, promising faster executions and coherent results without taking into account the consequence for storage.

3.1.2 Compression of VMs

VMs are used in many research domains. These are data that can be particularly massive, and often generated in large quantities. Therefore many projects have focused on how to compress them. The objective common to all the compression techniques presented hereinfater is to reach the best compression ratios.

The basic principle and the most straightforward technique to encode a VM is to use an indexed data structure: the list of all the vertex coordinates (three floating-point values, which amounts to 96 bits par vertex), followed by their connectivity. The connectivity is defined cell after cell, each cell being defined by the set of indexes of the adjacent vertices (8 integers per hex). They thus only provide estimates of an actual compression performance. To reduce the memory footprint or make the transmission of VMs faster, well-known techniques exist, for instance the *quantization* of vertex coordinates. It consists of constraining the vertex coordinates to a discrete and finite set of

values. Hence, it becomes possible to encode each coordinate with an integer index, instead of a 32-bit floating-point value. It is common to quantize the coordinates with 12 or 16 bits, reducing the geometry information by a compression factor of 2.60 or 2 respectively. Quantization inevitably introduces an irreversible loss in accuracy. Visualization typically tolerates precision loss (as long as visual *distortion* remains negligible), unlike some numerical simulations requiring a priori more precise computations.

Prediction (as well as related interpolation methods) further improves the geometry compactness. Predictive coding resorts to estimating the position of a vertex from already encoded neighbor vertices. *Prediction errors* (differences between predicted and actual positions) are generally smaller in amplitude and sparser, which makes their entropy coding (which codes differently frequently occurring patterns) efficient (Salomon and Motta, 2009, p. 63 sq.). Regarding the connectivity, when meshes are unstructured, the most frequent technique performs a traversal of mesh elements, and describes the incidence configurations with a reduced list of symbols. These symbols are then entropy coded. When meshes are structured, the connectivity is implicit, reducing its cost to zero. For such meshes, the only additional information to encode are geometrical discontinuities describing faults and gaps.

The basic tools previously presented can be implemented on the ontological structure of meshes, and improved in many different ways. Their combination, with the assistance of advanced compression techniques, permits more efficient tetrahedral or hexahedral mesh coding. Previously proposed algorithms are presented below, classified into two categories: *single-rate* and *progressive/multiresolution*.

Single-rate mesh compression They lead to a compact mesh representation, most of the time driven by efficient connectivity encoding. The first method for tetrahedral meshes, Grow & Fold, was presented by Szymczak and Rossignac (2000) at the end of the nineties. It is an extension of *EdgeBreaker* (Rossignac, 1999) developed for triangle meshes. The method consists in building a tetrahedral spanning tree from a root tetrahedron. The traversal is arbitrary among the three neighboring tets (*cf.* Subsection 2.1.2) of the cell currently processed, and 3 bits are needed to encode each cell. The resulting spanning tree does not retain the same topology as the original mesh, because some vertices are replicated during the traversal. "Fold" and "glue" techniques are thus needed during encoding to restore the original mesh from the tetrahedron tree. The additional cost is 4 bits, leading to a total cost of 7 bits per tetrahedron.

The *cut-border* initiated by Gumhold and Straßer (1998) was adapted to tetrahedral meshes (Gumhold et al., 1999). It denotes the frontier between tetrahedra already encoded and those to encode. At each iteration, either a triangle or an adjacent tetrahedron is added to the cut-border. In this case, if the added vertex is not already in the cut-border, this latter is included by a connect operation, and is given a local index. As the indexing is done locally, the integers to encode are very small, leading to a compact connectivity representation. In addition, two methods are proposed to encode geometry and associated properties, based on prediction and entropy coding. This method yields good bit rates (2.40 bits per tetrahedron for connectivity) for usual meshes, handles non-manifold borders, but worst-cases severely impact bitrates and runtimes (which tend to be quadratic).

Isenburg and Alliez (2003) are the first to deal with hexahedral VMs. The connectivity is encoded as a sequence of edge degrees — in a way similar to Touma and Gotsman (1998) for triangular meshes — *via* a region-growing process of a convex hull called *hull surface*. It relies on the assumption that hexahedral meshes are often highly regular, which implies that the majority of vertices are shared by 8 cells. It involves an almost constant edge degree all over the mesh, which significantly decreases the entropy of the connectivity information. A context-based arithmetic coder (Witten et al., 1987) is then used to encode the connectivity at very low bit rates, between 0.18 and 1.55 bits per hexahedron. Regarding geometry, a user-defined quantization first restricts the number of bits for coordinates, and then a predictive scheme based on the parallelogram rule encodes the position of vertices added during the region-growing process.

Krivograd et al. (2008) propose a variant to Isenburg and Alliez (2003) that encodes the vertex degrees — number of non-compressed hexahedra around a given vertex — instead of the edge degrees. On the one hand, this variant achieves better compression performances than Isenburg and Alliez (2003) for dense meshes. On the other hand, it only deals with manifold meshes, and the algorithm is complex as interior cells are encoded after border cells (it involves many specific cases to process when encoding the connectivity).

Lindstrom and Isenburg (2008) proposed an original algorithm for unstructured meshes called Hexzip. This algorithm is considered as fully lossless, because the initial indexing of vertices and hexahedra is preserved. For this purpose, connectivity is encoded directly in its indexed structure, by predicting the eight indices of an hexahedron from preceding ones. This technique is suitable because hexahedral meshes generally have regular strides between indices of subsequent hexahedra. A hash-table is then used to transform the index structure into a very redundant and byte-aligned list of symbols, that can be compressed efficiently with gzip (Deutsch, 1996). Concerning the geometry, spectral prediction (Ibarria et al., 2007) is used. This algorithm is faster and less memory intensive than Isenburg and Alliez (2003) as the connectivity is not modified. It handles non-manifold meshes and degenerate elements. On the other hand, bitrates are higher because of the lossless constraints.

Unlike methods presented above, Chen et al. (2005) focus on geometry compression for tetrahedral meshes. The authors proposed a *flipping* approach based on an extension of the *parallelogram rule* (initially proposed for triangle meshes (Touma and Gotsman, 1998)) to tetrahedra. It consists in predicting the position of an outer vertex of two face-adjacent tetrahedra, with respect to the other vertices. To globally optimize the geometry compression, a Minimum Spanning Tree minimizing the global prediction error for the whole mesh is computed. This method is more efficient than prior flipping approaches whose traversal does not depend on the geometry, but solely on the connectivity.

Streaming compression is a subcategory of single rate compression, dedicated to huge data that cannot fit entirely in the core memory. A particular attention to I/O efficiency is thus required, to enable the encoding of huge meshes with a small memory footprint. Isenburg and coworkers are the first to propose streaming compression for VMs (extended from his method for triangular meshes): for tetrahedral meshes (Isenburg et al., 2006), and then for hexahedral meshes (Courbet and Isenburg, 2010). In the latter, for instance, the compressor does not require the knowledge of the full list of vertices and cells before encoding. The compressor starts encoding the mesh as soon

as the first hexahedron and its eight adjacent vertices have been read. For a given hexahedron: i) its face-adjacency is first encoded in function of its configuration with hexahedra already processed; ii) positions of vertices that are referenced for the first time are predicted (spectral prediction from adjacent cells); iii) prediction errors are encoded; iv) data structures relative to vertices, becoming useless (because their incidence has been entirely described) are finally removed from memory. Compared to other single rate techniques, streaming tends to achieve similar compression performances for geometry, but poorer performances for connectivity.

Progressive/multiresolution mesh compression *Progressive* algorithms (also called scalable or *multiresolution*) enable the original meshes to be represented and compressed at successive LODs (levels of details). The main advantage is that it is not necessary to decompress a mesh entirely before vizualising it. A coarse approximation of the mesh (also known as its lowest resolution) is first decompressed and displayed. Then this coarse mesh is updated with the successive LOD (termed higher resolutions) that are decompressed progressively. While they cannot achieve yet compression performance of single-rate algorithms, progressive algorithms are popular because they enable LOD, and also adaptive transmission and displaying, in function of user constraints (network, bandwidth, screen resolution...).

Pajarola et al. (1999) are the first to propose in 1999 a progressive algorithm dedicated to VM compression. This work is inspired by a simplification technique for tetrahedral meshes (Staadt and Gross, 1998). It simplifies a given tetrahedral mesh progressively, by using successive *edge collapses* (Hoppe et al., 1993). Each time an edge is collapsed, its adjacent cells are removed, and all the information required to reverse this operation is stored: index of the vertex to split, and the set of incident faces to "cut". Thus, during decompression, the LODs can be also recovered iteratively, by using the stored data describing *vertex splits*. During coding, an edge is selected such as its collapse leads to the minimal error, with respect to specific cost functions. This algorithm gives a bitrate inferior to 6 bits per tetrahedron (for connectivity only).

In 2003, Danovaro et al. (2002) propose two progressive representations based on a decomposition of a field domain into tetrahedral cells. The first is based on *vertex splits*, as the previous method, the second is based on *tetrahedron bisections*. This operation consists in subdividing a tetrahedron into two tetrahedra by adding a vertex in the middle of its longest edge. Unlike with vertex splits, the representation based on *tetrahedron bisections* is obtained by following a coarse-to-fine approach, *i.e.*, by applying successive bisections to an initial coarse mesh. Also, this representation only needs to encode the difference vectors between the vertices added by bisections and theirs real positions. This representation is thus more compact, as the mesh topology does not need to be encoded, but it only deals with structured meshes.

VMs multiresolution decomposition based on wavelets (Jacques et al., 2011) was proposed by Boscardín et al. (2006) for tetrahedral meshes. It is based on the tetrahedron subdivision scheme (Bey, 1995) that transforms a tetrahedron into 8 sub-tetrahedra, by introducing 6 new vertices on each edge. After analysis, the input mesh is replaced by a base tetrahedral mesh, and several sets of detail wavelet coefficients. Although coefficients corresponding to differences between two resolu-

tions could be encoded for mesh synthesis, this work does not provide an actual compression scheme.

Chizat (2014) proposed a prototype for a multiresolution decomposition of geoscientific hexahedral meshes with the pillar grid structure (*cf.* Subsection 2.1.2). His main contribution resides in a multiresolution analysis (MRA) that partially manages geometrical discontinuities representing the faults. It can be achieved by using a morphological wavelet transform (*cf.* Subsection 3.2.1). This non-separable transform enables the preservation of some fault shapes at different resolutions, as depicted in Figure 3.1.

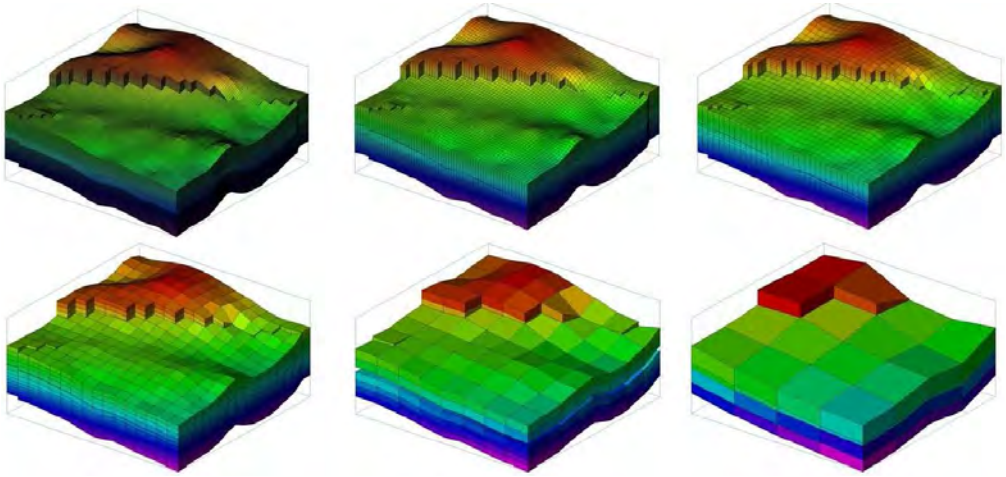


Figure 3.1: Dyadic non-separable multiresolution rendering on a simple geological mesh Chizat (2014).

This work is the first to deal with meshes from geosciences at IFP Energies nouvelles. The multiresolution concept manages structural discontinuities, while preserving the fault network through resolutions. The proof of concept is developed in Peyrot et al. (2019), and compression performance is thoroughly analyzed in (Bouard et al., 2018). HexaShrink is detailed later in Section 3.2.

3.1.3 Compression of scientific data

Ongoing advances in computer science allows performing simulations on ever larger and detailed models, while their management becomes ever more difficult. This subsection present solutions for compressing $3D$ experimental and simulation data. By using various approaches: data reduction (Cunningham and Ghahramani, 2014), data clustering (Geist and Reed, 2016) and data indexation (Kraska et al., 2018). Recently Li et al. (2018) and Duwe et al. (2020) published surveys on compression of scientific data in the context of HPC, which indicates a growing interest for this topic. Note that a major driving application for compression concerns climatology-related simulations (meteorology, oceanography), where the data generated tend to be the heaviest ones (*cf.* Subsection 4.1.1). As always in compression, two categories can be identified: lossless and lossy.

Lossless techniques Generic lossless methods are used by broad audience to compress a wide variety of data, regardless of data features (dimension, type, ...). Mainstream methods although generic, such as gzip or LZMA (tested in Subsection 3.4.1), may be efficient in term of compression ratios and execution times. Nevertheless, lossless methods dedicated to scientific data are now available.

FPC (Floating-Point Compression) is a fast and lossless algorithm conceived to optimize the transmission and the storage of floating-point numbers Ratanaworabhan et al. (2006). Based on prediction techniques, FPC compresses better and faster than other algorithms available at the time of its release. But its efficiency decreases with the randomness rate of the data.

Later, many scientists, particularly climatologists, adopted two standard formats to store multi-dimensional data in an unified library interface: NetCDF (Network Common Data Form) and HDF5 (Hierarchical Data Format 5) (Koranne, 2011). Yet, in this format the original data are not available anymore, and must be decompressed entirely to be used. HDF5 provides compression components to develop tools designed specifically for climatology, such as MAFISC (Hübbe and Kunkel, 2012). A sequence of filters is applied to the data and completed by generic encoding tools.

Other compression formats have been developed specifically for HPC. For instance, ISOBAR (Schendel et al., 2012) proposes to identify and filter hard-to-compress datasets in order to better target the compression needs.

Lossy techniques Lossless methods ensure a perfect reconstruction of the initial data, but their compression performance is limited. Despite this limitation, many researchers are reluctant to use lossy compression, fearing that the losses will bias or distort simulation results. However, as highlighted ironically by Kipnis et al. (2018), "data discretization in simulation is already a form of quantization, which leads to a lost of information".

Today, popular formats such as HDF5 progressively incorporate powerful and easily implementable lossy compression tools (Linear Packing, Layer Packing (Silver and Zender, 2017), Bit shaving, Bit Grooming (Zender, 2016), Digit Rounding (Delaunay et al., 2019) *etc.*) that remove last bits of values considered as noise and/or irrelevant data. These irreversible and drastic components are completed by lossless techniques that rearrange bits (Shuffle) and coders (DEFLATE, Zstandards). The results obtained by Delaunay et al. (2019) on meteorological and oceanic experimental datasets made of floating values are promising. Their degradation is positively assessed by objective metrics (SNR, mean relative error, mean absolute error).

Lossy compression allows achieving higher compression gains, but implies to control the data degradation strictly. However, there is no metric established in the context of simulation, because expectation vary according to the application. In this case, quality metrics borrowed from the visualization field are found sufficient, and are used to appreciate the distortion involved by lossy compression (maximum point wise error, root mean squared error (RMSE), peak signal to noise ratio (PSNR)).

Lossy compression is now a common place in climatology (Baker et al., 2017, 2019), where the data volume for the forecast are the most massive. One of our oldest references in simulation data compression ((Bradley and Brislawn, 1993)) already dealt with storage of data generated by supercomputers from NCAR.

As already noted in the context of lossless compression, it is fairly common to test popular lossy compression methods on scientific data although not adapted. JPEG and JPEG2000 developed for multimedia data have been for example tested on climatology by (Hübbe et al., 2013; Woodring et al., 2011; Baker et al., 2014), and fluid mechanics (Schmalzl, 2003).

Computational fluid dynamics (CFD) is powerful tool for simulating turbulent flows. However, computational time can be high, due to the large scale of simulated data. To address this problem, Nakahashi (2005) proposes a compact support called BCM (Building Cube Method) to decrease the number of cells. It assemblies sub-cubes (Cartesian grids) at variable resolutions to create an unstructured grid, with local refinement adapted to geometry and flow features. To diminish the size of binary data and relieve the computation, run-length coding is applied to the BCM grids. The work of Sasaki et al. (2015) improves this method by applying the filter CDF 9/7 to each cube. Wavelet coefficients are then quantified at various thresholds, and then zerotree coded (the method is presented in Subsection 3.5.2).

In Clyne et al. (2007), VAPOR enables compression and visualization of massive computational datasets. It also facilitates analysis thanks to its interface. This work has been motivated by the data complexity of two particular applications: turbulent plume dynamics and current sheet formation. VAPOR is based on a wavelet decomposition (mainly Haar) that generates a hierarchical representation. Last version of VAPOR (Li et al., 2019) guarantees that the "visualization package [is] tailored to analyze simulation data in earth system".

To further investigate lossy compression based on wavelets, Li et al. (2015) worked on a volume data set characterizing turbulent flows. The efficiency of several wavelets (Haar, CDF 9/7, CDF 8/4) are evaluated for visualization, with classical objective and proper field metrics: the best option seems to be a combination of the filter CDF 9/7 and coefficient prioritization (instead of approximation). The method has been also implemented on spatio-temporal data (Li et al., 2017b).

Wavelet based methods inspired many works on scientific data compression in various fields, from seismic in geosciences to tomography in medical. Most of the works deals with experimental data, but very few with simulation data. However, in recent years, simulation fields tended to catch up. As mentioned above with our oldest reference on the topic, we observed that the interest of integrating compression in a simulation workflow is not new. Bradley and Brislawn (1993) worked on wavelet based solution with quantization step for NCAR ocean models. Its dimensions were limited, but already problematic to repatriate the simulated data from supercomputers. Hereafter open source libraries as QccPack Fowler (2000) permit simple implementations of methods based on wavelets. Coming back to fluid mechanics, we can mention other works using wavelets on various structured simulation data, from $2D$ to $4D$ (Wilson, 2002; Schmalzl, 2003; Kolomenskiy et al., 2018) or point clouds (Salloum et al., 2018). Loddoch and Schmalzl (2006) proposed an equivalent approach for compressing volume fluid dynamic datasets by replacing wavelets by other transform and such as DCT, B-splines, *etc.* and predictors such as Lorenzo, *etc.*

Fpzip (Lindstrom and Isenburg, 2006) is an online compressor for floating-point values, implementable on the I/O of diverse applications. It manages various data through its multidimensional prediction scheme based on the Lorenzo predictor. Depending on the usage, the scheme proceeds

progressive lossy compression by truncation of the less significant bits. The impact of such a reduction is notably studied on different simulation codes (LULESH: Lagrangian shock hydrodynamics, Miranda: High-order Eulerian hydrodynamics, details in Subsection 4.1.1) by Laney et al. (2014) as detailed in Subsection 4.1.2.

ISABELA (In-situ Sort-And-B-spline Error-bounded Lossy Abatement) by Lakshminarasimhan et al. (2011) provide an adaptation to noisy data. To prepare it, a smoothing spatial preprocessing is applied. Then, the data is approximated using B-splines or wavelets. Additionally, the temporal dimension is also exploited by temporal pattern identification. However, the decompression can not be performed on the entire data, but only on subsets, which complicates its usage.

Iverson et al. (2012) develop a compression model for unstructured meshes in geosciences, by conversion of the grid into a graph, taking advantage of the locality. In such a structure, the graph is defined by N nodes of the mesh, the set of edges E connecting two nodes. A zero weight indicates if the connection does not exist, otherwise a weight proportional to the connection importance is used. It can correspond to a distance if considering the mesh geometry, but can deal with multiple other properties.

After fpzip, Lindstrom (2014) introduces ZFP, available on HDF5, ADIOS library. The concept is notably inspired by fixed-rate texture compression methods used in graphics hardware. It permits a random access to compressed floating-point data, by splitting into blocks at variable precision. Floating-point representation is exploited by differentiating the exponents and the mantissa in IEEE-754 format. DCT is applied to the mantissa elements within subblocks, and the resulting coefficients are zerotree coded. The exponents are distinctly stored. A current work aims at determining a closed form expression for bounds on the error introduced by the three compression options of ZFP (Diffenderfer et al., 2019).

SZ (Cappello et al., 2019) is a predictive method for multidimensional floating point data. A first step consists in flattening the data into a single dimensional array. If appropriate, a log-mapping transform can be applied (in the version 2.0 (Liang et al., 2018a)). The array is then read progressively, and the values are predicted from the previous ones, looking for the best fitting among various deterministic models (mean-integrated Lorenzo predictor, curve fitting scheme constant, linear or quadratic). Quantization, Huffman coding and gzip complete the compression workflow.

In two years, Ainsworth et al. (2018, 2019, 2020a,b) produce four sequential works on multilevel techniques for compression and reduction of scientific data (*MGRAD* project). From univariate to multivariate cases, Ainsworth et al. (2020b) finally manages 2D and 3D meshes. This method employs an orthogonal decomposition based on the Riesz basis properties. In this study, airfoils are studied to evaluate preservation of aerodynamic force and coefficients of pressure. Other applications are mentioned in Subsection 4.1.2 to evaluate the impact of compression in simulation.

Discussion SZ and ZFP appear today as the most competitive compression tools for scientific data. Complementary tools are trying to exploit their shortcomings. For example, FRaZ (Underwood et al., 2020) works on improvement of user modes (fixed-rate, absolute error bound *etc.*) and Wang

et al. (2019) are interested in intrinsic metrics of both compression tools, to better understand processes.

In addition to netCDF and HDF, new libraries specific to lossy compression emerge for large scientific data. For example, SCIL (Scientific Compression Library) (Kunkel et al., 2017) notably integrates ZFP and SZ. Several compression components can be combined according to the needs of users. CubismZ, created by Hadjidoukas and Wermelinger (2019) is another recent example. This library proposes a parallel implementation of methods based on wavelets, or tools such as ZFP, SZ and fpzip.

Finally, lossy compression has been often employed in the context of simulation, but researchers remain cautious about the risk of data *degradation*. Beyond the size reduction, compression promotes discussions toward a rationalization of ever growing datasets, on terms of "required" precision and discretization. Even if compression did not seem to pervade the field of simulation in geosciences (except in Iverson et al. (2012) and Lindstrom et al. (2016)), an awareness on the need to manage data at a reduced precision already exists, if considering the simplification brought by upscaling.

3.1.4 How taking advantage of these three techniques for a "dream representation" of GVM?

We now discuss the main features borrowed to the methods of upscaling/upgrading, VM compression and scientific data compression. Then, we set the bases of a new representation adapted to our GVMs. Concerning our "dream representation", we can gather the following key observations:

- Our representation has to be based on an **embedded multiscale decomposition scheme**, which offers obvious advantages for visualization, storage, but also simulation. One of the avantages is that the decomposed data could not take up more space than the initial data, unlike the upscaled/upgraded data.

- Despite their high performance in term of compression, the progressive methods developed for VMs are not appropriate for our data. Indeed, the GMs contain fault networks, involving local geometrical discontinuities that are not managed by these generic compression methods. Therefore **our multiscale mesh representation must take into account the specific morphological features of the initial data, and must preserve them as much as possible across the resolutions.**

- The continuous properties are $3D$ floating values, sometimes of **high numerical precision**, which is not always necessary for visualization or simulation. Thus **our representation must propose an encoding of these properties at refinable precision**, according to the application and the user needs. To assess the efficiency of our encoder at refinable precision, popular compression tools for scientific data such as ZFP or SZ will serve as reference for comparison study.

- Particular orientations are observable for properties, that can be explained by their geological origin. **This anisotropy needs to be preserved at various scales**, preserving by this way the coherency of the mesh in a multiresolution structure.

During a previous collaboration between IFP Energies nouvelles and I3S, some of these observations have been considered to implement the core of HexaShrink (Peyrot et al., 2016, 2019). HexaShrink is a multiscale decomposition scheme for hexahedral meshes from geosciences. It combines several wavelet transforms adapted to the specific features of these meshes (detailed in next section). Firstly developed for multiscale renderings, the HexaShrink structure is the starting point of our current research, focused on compression and its impact on simulation results.

3.2 Description of HexaShrink

This section introduces HexaShrink, the mesh hierarchical structure on which our work has been based. Some paragraphs of this section are extracted from Peyrot et al. (2019).

MRA or multiscale *approximation* can be interpreted as a decomposition of data at different resolutions, LOD or scales, through a recursive *analysis* process. It is called exact, reversible or invertible when a *synthesis* scheme can retrieve the original data. Inter-scale relationships (Chaux et al., 2007, 2008) often yield *sparsification* or increased *compressibility* on sufficiently regular datasets. In discrete domains, each *analysis* stage transforms a set of values (continuous or categorical, in one or several dimensions), denoted by S^0. The resulting representation consists in one subset S^{-1} that approximates the original signal at a lower resolution, *plus* one subset of details D^{-1}, or a combination thereof. The latter represents information missing in the approximation S^{-1}. Depending on the MRA scheme, the lower *resolution* S^{-1} may represent a coarsening or "low frequencies" of the original samples, or an upscaling in geosciences (*cf.* Subsection 3.1.1). The subset D^{-1} represents refinement, fast variation or "high-frequency" details removed from S^0. We consider here exact systems, allowing the perfect recovery of S^0 from a combination of subsets S^{-1} and D^{-1}. Hence, a similar analysis stage can be applied iteratively, and perfectly again, to the lower resolution S^{-1}, in a so-called pyramid scheme. Thus, with the non-positive extremum decomposition level L, and indices $0 \geq l \geq L$, after an $|L|$-level multiresolution decomposition, the input set S^0 is now decomposed and represented by the subset S^L — a (very) coarse approximation of S — and $|L|$ subsets of details $D^L, \ldots, D^l, \ldots, D^{-1}$, representing information missing between each two consecutive approximations.

To get an idea of what a MRA will generate from VMs, Figure 3.2 gives an overview of a decomposition obtained with HexaShrink on one of our GMs. To obtain this decomposition, we consider in the following four different MRA flavors, all called wavelets for simplicity. They stem from iterated, (rounded) linear or non-linear combinations of coefficients, as well as separable (applied separately in $1D$ on each direction) or non-separable ones. Without going into technicalities here (*cf.* Subsection 3.2.1), computations are performed using the *lifting scheme*. It suffices to mention that lifting uses complementary interleaved grids of values, often indexed with odd and even indices. Values on one grid are usually predicted (approximations) and updated (details) from the others. The main interests reside in reduced computational load, in-place computations and the possibility to maintain exact integer precision, using for instance only dyadic-rational coefficients (written as $m/2^n$, $(m, n) \in \mathbb{Z} \times \mathbb{N}$) and rounding. We refer to Jacques et al. (2011, sections 2.3., 3.2 and 4.3) for a concise account on both non-separable and non-linear wavelet MRAs, and to Sweldens (1996); Bruekers and van den Enden (1992); Rao and Bopardikar (1998); Kovačević et al. (2012) for more comprehensive vision of wavelets and their lifting implementations. A recent use in geological model

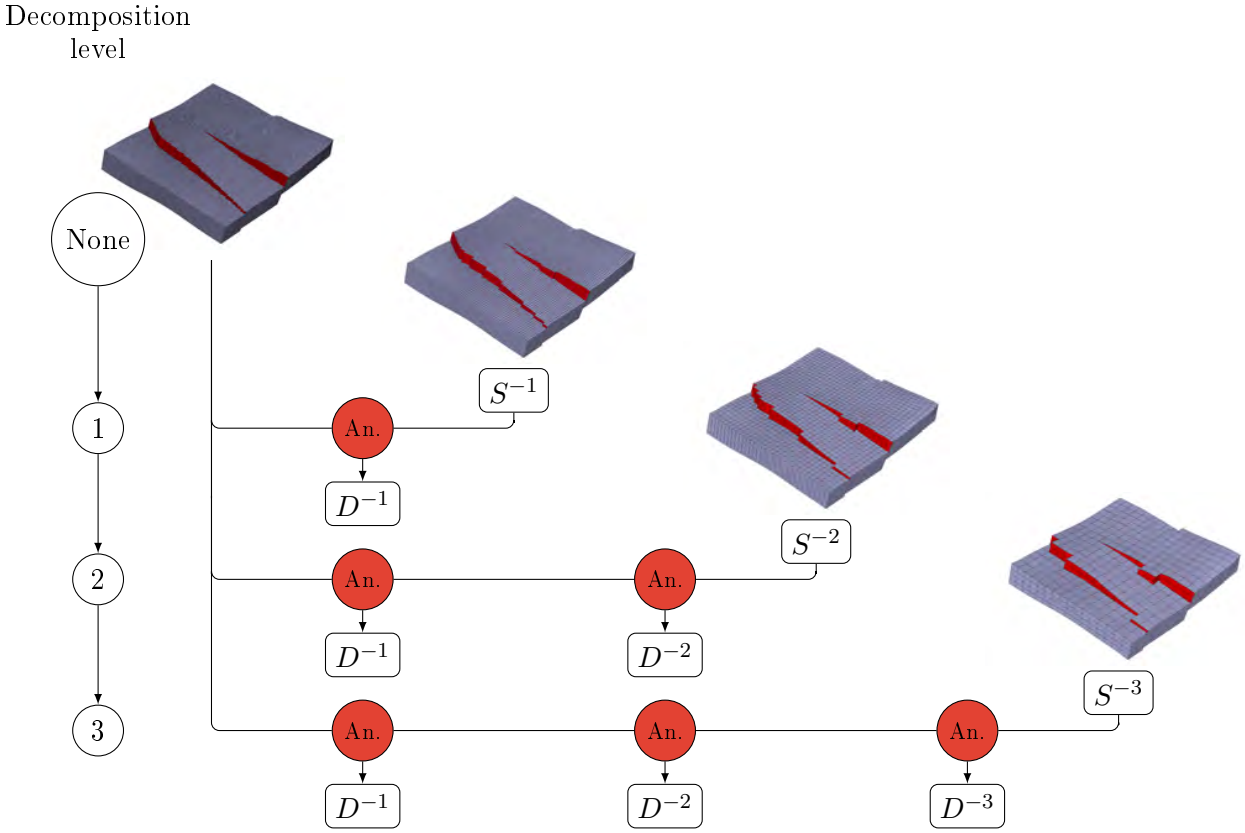


Figure 3.2: A $|L|$-level decomposition of a geological mesh obtained by MRA (mesh#6 from Figure 3.5). From the initial mesh (top-left), an approximated version of the initial mesh (S^L) plus $|L|$ sets of details (D^i) are obtained. The number of sets of details increase with the level of decomposition $|L|$, but the global quantity of data remain stable.

upscaling is given in Rezapour et al. (2019).

More simply put, for our hexahedral VMs, the dyadic analysis stage transforms each cell block $\mathcal{C}^l$ of values around $2^3 = 8$ contiguous cells (possibly borrowing values from a limited cell neighborhood) at resolution l. They are turned into 1 *approximating cell* (lower resolution (S^{l-1}), and a subset of $2^3 - 1 = 7$ detail cells D^{l-1}, as depicted in Figure 3.3 along with the reverse synthesis stage.

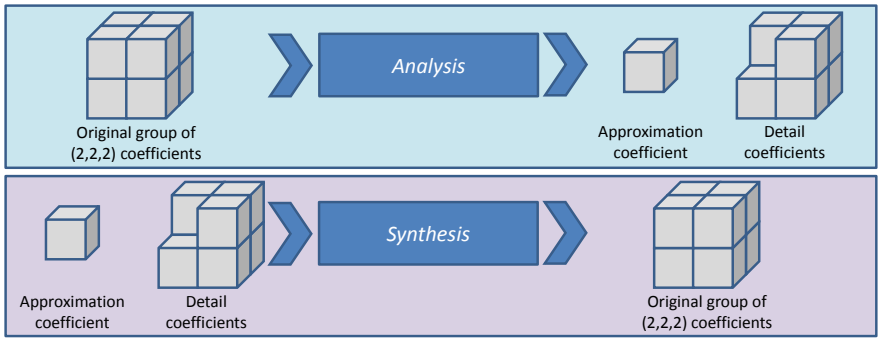


Figure 3.3: Analysis and synthesis stages for VMs.

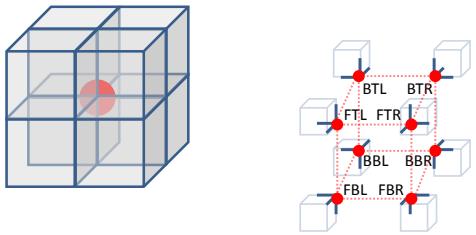
Hence, if a VM at resolution l is composed of $\left(\mathcal{C}_i^l \times \mathcal{C}_j^l \times \mathcal{C}_k^l\right)$ cells, $\mathcal{C}_i^l$ being the number of cells in direction i, the VM of lower resolution will be of dimension $\left\lceil \frac{\mathcal{C}_i^l}{2} \right\rceil \times \left\lceil \frac{\mathcal{C}_j^l}{2} \right\rceil \times \left\lceil \frac{\mathcal{C}_k^l}{2} \right\rceil$, to take into account non-power-of-two sized grids. As several digital attributes are associated to each cell (geometry, continuous or categorical properties), different types of MRA are performed separately on the different variables defining these properties, as explained in the following sections.

3.2.1 Multiresolution scheme for geometry

Standard linear MRA schemes rely on smoothing or averaging and difference filters for approximations and details, respectively. To preserve coherency of representation of geometrical discontinuities — whatever the resolution — a special care is taken to avoid excessive smoothing, while at the same time allowing the reverse synthesis. As the pillar grid format is used (*cf.* Subsection 2.1.2), vertices are inevitably positioned along pillars. So, our multiresolution scheme for geometry information only focuses on:

- the z coordinates of the 8 vertices associated to each node. According to the naming convention presented in Figure 3.4, those 8 vertices can be differentiated according to their relative positions [Back (B)/Front (F), Bottom (B)/Top (T), Left (L)/Right (R)];
- the x and y coordinates of the nodes describing the low (bottom) and high (top) extremities of all the pillars (the x and y coordinates of intermediary nodes being implicit). The nodes are called hereinafter the *floor* and *ceil* nodes, respectively.

The GVMs can exhibit very irregular boundaries. Hence, a Boolean field called ACTNUM may be associated to each cell to inactivate its display (and its influence during simulations as well). It



(a) A node and its 8 surrounding cells.

(b) Splitting view of the node into its 8 vertices.

Figure 3.4: Vertex naming with the Back (B)/Front (F), Bottom (B)/Top (T), Left (L)/Right (R) convention.

enables the description of either mesh boundaries (Figure 3.5), or caves/overhangs. Resultantly, this Actnum field must be carefully considered during the MRA of the geometry information, to avoid artifacts at lower resolutions on frontiers between active and inactive cells (*cf.* Subsection 3.2.1 and Figure 3.10).

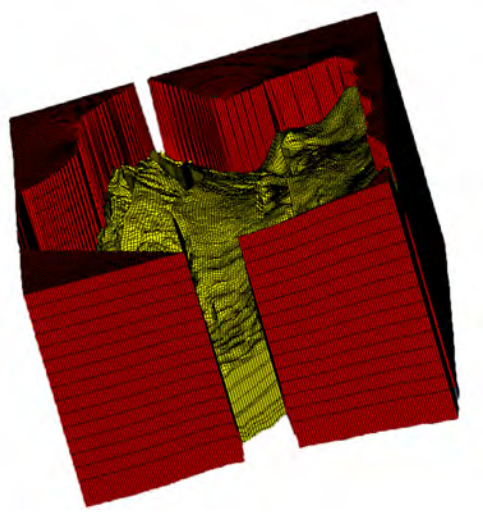


Figure 3.5: Mesh#5 (in yellow) has inactive cells (in red) to describe its boundaries using the Actnum field.

By construction, most GVMs have no horizontal fault, as there is no vertical gap between any two adjacent layers of cells. For every node, each of the four top vertices has the same z coordinate as its counterpart bottom vertex. Therefore, from now on, our geometry multiscale representation method only deals with the z coordinates of the bottom vertices BBL, BBR, FBL, and FBR of each node.

An instance of the decomposition shown in Figure 3.3 can be implemented with the proposed two-step technique depicted by arrows in Figure 3.6:

- A non-linear and non-separable $2D$ **morphological wavelet transform** applied on the nodes, in order to detect the faults in the input VM, and then to preserve their coherency in the lower resolutions. This step relies on a **fault segmentation** within the input VM obtained by studying all possible fault configurations for the top view of the VM (see Figure 3.7);
- A non-linear $1D$ **wavelet transform** applied on the output of the above first step to analyze **the z coordinates of the vertices** along each pillar. The same $1D$ wavelet transform is also applied on the sets of x and y coordinates of the *floor* and *ceil* nodes, to complete the "horizontal" decomposition.

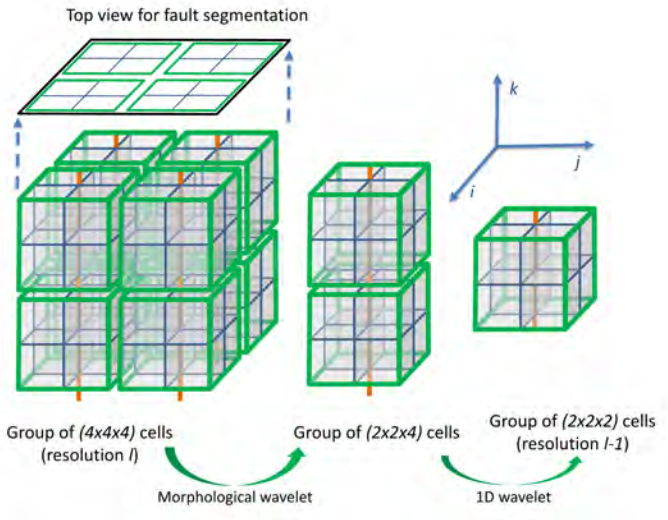


Figure 3.6: HexaShrink multiresolution scheme for geometry: (left) input grid and its top view; (middle) output from the non-separable, non-linear $2D$ morphological wavelet based on a fault segmentation (based on the top view); (right) non-linear $1D$ wavelet transform along pillars (orange lines).

Fault segmentation This stage detects the faults in the original mesh, in order to preserve them during the morphological wavelet analysis. For each node, a dozen of fault configurations, depending on BBL, BBR, FBL, and FBR, is possible: fault-free (1), straight (2), corner (4), T-oriented (4) or cross (1), as illustrated in Figure 3.7.

Each configuration depends on the four orientations of the cardinal axes (north, south, east and west), which are either active or inactive. For instance, the T-north configuration has its south axis inactive, while the three remaining ones are active. Assuming that a fault configuration is z-invariant, meaning that the nodes belonging to the same pillar present the same fault configuration,

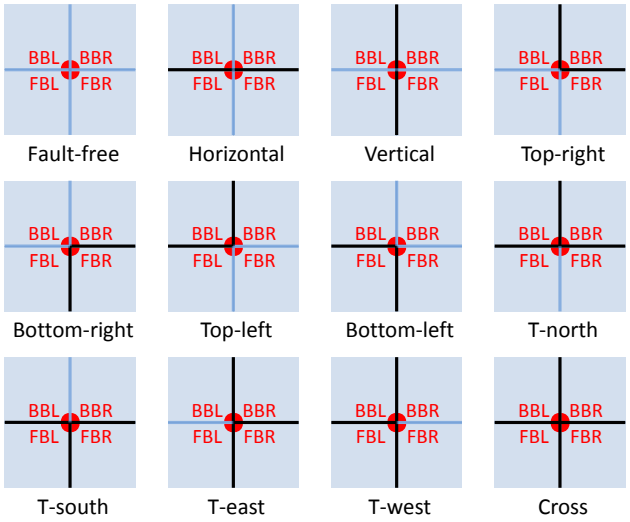


Figure 3.7: The 12 possible fault configurations (in black lines) at a given node.

a single $2D$ configuration map is sufficient to represent the fault configuration of the whole mesh, as illustrated by Figure 3.8.

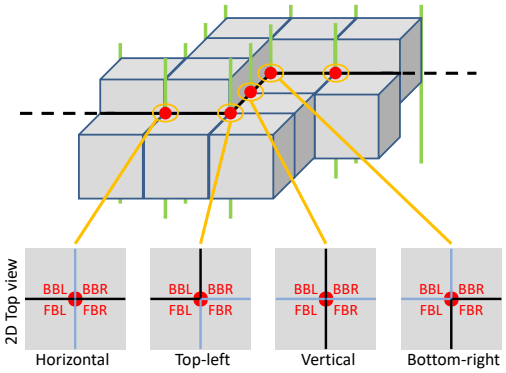


Figure 3.8: Fault segmentation within the original mesh.

Horizontal $2D$ morphological wavelet transform The fault segmentation guides the MRA to preserve faults, as much as possible, all over the decomposition process. The fault configuration of 4 associated nodes at resolution l is used to predict the extension of the downsampled fault structure at resolution $l-1$.

This horizontal prediction is based on the logical function OR ($\vee$), computed on each side of each group of 4 nodes. For instance, a resulting fault node configuration contains a west axis if the fault configurations of the 2 left nodes contain at least 1 west axis, as illustrated in Figure 3.9. By repeating the procedure for each axis of each resulting node, fault node configurations at lower resolutions are fully predicted. This non-linear and peculiar choice is meant to maintain a directional

flavor of orientated faults for flows; other choices could be devised, depending on physical rules and geological intuitions.

Finally, from this prediction, the node whose configuration minimizes its distance with the predicted one, corresponds to the aforementioned approximation coefficient, which will be part of the novel Z matrix at lower resolution $l-1$. The same procedure can be applied recursively until the wanted resolution.

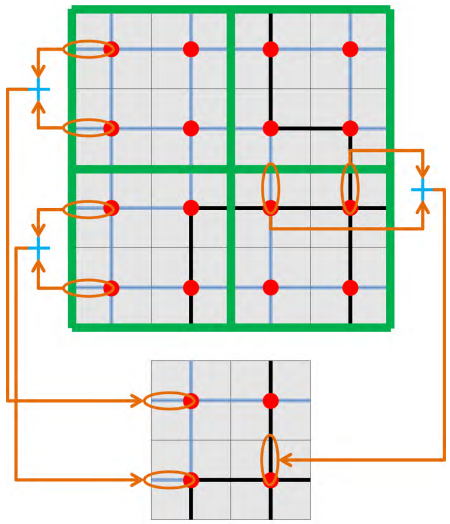


Figure 3.9: Prediction of a fault node at resolution $l-1$ from the four parents' configuration at resolution l, orange ovals denoting $\vee$ operands.

Rounded linear $1D$ wavelet transform This $1D$ wavelet transform is applied on the output of the above horizontal $2D$ morphological wavelet, to analyze the z coordinates of the 4 sets of vertices BDR, FDR, BDL and FDL separately, along each selected pillar. The HexaShrink multiresolution here acts on z-locations, decomposing them at each scale location z^l into a subsampled pillar coordinate z^{l-1} and its associated detail d^{l-1}. By geomodel construction, coordinate behavior along the pillars is expected to be relatively smooth. This entails the use of a modified, longer spline wavelet. The latter can be termed LeGall (Le Gall and Tabatabai, 1988), or CDF 5/3 (after Cohen, Daubechies and Feauveau (Cohen et al., 1992)), or biorthogonal 2.2 from its vanishing moments.

The lifting analysis operations *Prediction* and *Update* are depicted by Figure 3.6. To retrieve respectively the sets of details d^l and the z^l coordinates at resolution $l-1$ from scale l, the following equations are used ($\forall n \in \mathbb{N}$):

$$d^{l-1}[n] = z^l[2n+1] - \left\lfloor \frac{z^l[2n] + z^l[2n+2]}{2} \right\rfloor , \tag{3.1}$$

$$z^{l-1}[n] = z^l[2n+0] + \left\lfloor \frac{d^{l-1}[n-1] + d^{l-1}[n]}{4} \right\rfloor , \tag{3.2}$$

where both dyadic integers and rounding are evident. With rounding, lifting schemes can thus manage integer-to-integer transformations Calderbank et al. (1998). For synthesis, to reconstruct

resolution l from resolution $l-1$, we only have to reverse the order and the sign of the equations:

$$z^l[2n] = z^{l-1}[n] - \left\lfloor \frac{d^{l-1}[n-1] + d^{l-1}[n]}{4} \right\rfloor , \tag{3.3}$$

$$z^l[2n+1] = d^{l-1}[n] + \left\lfloor \frac{z^l[2n] + z^l[2n+2]}{2} \right\rfloor . \tag{3.4}$$

Managing externalities: borders and boundaries A pertinent multiresolution on complex meshes requires to cope with externalities that may hamper their handling: floor and ceil borders and outer boundaries (Figure 3.5). First, to keep borders unchanged from the original mesh, throughout all resolutions, the following constraints must be met:

$$z^{l-1}[0] = z^l[0] , \tag{3.5}$$

$$z^{l-1}[n_k^{l-1} - 1] = z^l[n_k^l - 1] . \tag{3.6}$$

Both constraints can be fulfilled if one satisfies the following conditions:

- Floor border condition to meet (3.5):

$$d^{l-1}[-1] = -d^{l-1}[0] , \tag{3.7}$$

- Ceil border condition to meet (3.6):

$$d^{l-1}[n_k^{l-1} - 1] = -d^{l-1}[n_k^{l-1} - 2] , \quad (n_k^l \text{ odd}) \tag{3.8}$$

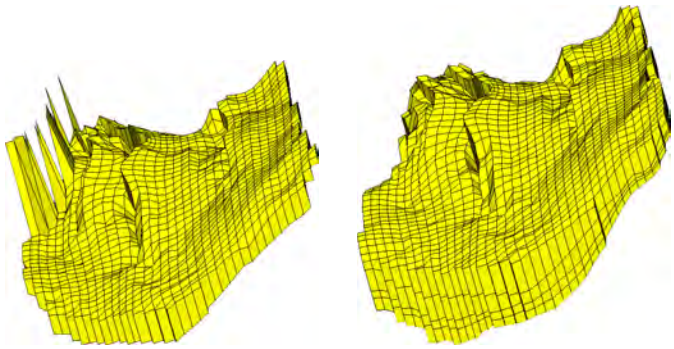
$$d^{l-1}[n_k^{l-1} - 1] = -d^{l-1}[n_k^{l-1} - 2] \quad (n_k^l \text{ even}) \\ +4z^l[n_k^l - 1] - 4z^l[n_k^l - 2] . \tag{3.9}$$

To complete the MRA of the geometry, the same rounded $1D$ wavelet is also applied to the sets of *floor* and *ceil* nodes of the initial VM, to get the x and y coordinates of the extremities of the remaining pillars at the lower resolution.

Second, the Actnum field should also be considered to lessen mesh boundary artifacts. Indeed, severe disturbances may appear at lower resolutions if not wisely processed during analysis, as shown in Figure 3.10. A cell is deemed active if and only if its 8 adjacent vertices are active at the resolution l. During our study, we found that one vertex at resolution $l-1$ could be considered active if and only if its parent vertices selected by the morphological wavelet at resolution l (*cf.* Subsection 3.2.1) are active. So, a cell at resolution $l-1$ is considered active if and only if its 8×2 corresponding parent vertices are active at resolution l.

3.2.2 Multiresolution scheme for properties

Continuous properties Once the geometry is coded, one can focus on associated continuous properties. For scalar ones, a value $p_i \in \mathbb{R}$ is associated to each cell i in the mesh. Consistently with



(a) Boundary artifacts without proper ACTNUM management.

(b) Lower mesh resolution with efficient ACTNUM Boolean values care-taking.

Figure 3.10: Inadequate ACTNUM fields management during analysis may lead to severe boundary artifacts (left) that can be dealt with (right) as *examplified* with mesh#5 from Figure 3.5 .

the handling of cell blocks $\mathcal{C}$ of $2\times2\times2$ cells throughout scales, we use an adaptation of the well-known Haar wavelet. The resolution $l-1$ is a scaled average of cells at resolution l. The approximation coefficient p^{l-1} is thus the average value of the related eight property coefficients $\{p^l_1, p^l_2, \ldots, p^l_8\}$. The seven details required for synthesis are differences with respect to the approximation coefficient:

$$p^{l-1} = \frac{1}{8}\sum_{n=1}^{8} p^l_n \,; \quad d^{l-1}_n = p^l_n - p^{l-1} \,, \forall n \neq 1 \,.$$

To deal with real-valued (floating-point) properties, and avoid accuracy imprecision due to the divide operator, we introduce the following modifications. First, reals are mapped into integers up to a user-defined precision, here with a 10^6 factor. Second, we disable the division by using a sum. The analysis system thus becomes:

$$p^{l-1} = \sum_{n=1}^{8} p^l_n \,; \quad d^{l-1}_n = 8p^l_n - p^{l-1} \,, \forall n \neq 1 \,,$$

and the synthesis system turns into:

$$p^l_n = \frac{1}{8}(d^{l-1}_n + p^{l-1}) \,, \forall n \neq 1 \,; \quad p^l_1 = p^{l-1} - \sum_{n=2}^{8} p^l_n \,.$$

Approximation and coefficients are stored as is. To recover the accurately scaled values, the division operator should however be applied as a simple linear post-processing.

Categorical properties We complete the global mesh multiresolution decomposition with an original categorical-valued scheme called *modelet* (Antonini et al., 2017). We assume that a mesh cell category belongs to a set of classes $\Omega_0 = \{\omega_1, \omega_2, ..., \omega_W\}$, taking discrete values. The cell block

$\mathcal{C}^l = \{p_1^l, p_2^l, \ldots, p_8^l\}$ thus contains, at resolution l, integers indexing categories from Ω_l. They take values in a subset of Ω. The multiresolution scheme is expected to produce, at lower resolutions, discrete values in embedded subsets: $\Omega_0 \supset \Omega_{-1} \supset \cdots \supset \Omega_l \supset \cdots$. In other words, a cell category can only belong to an existing category at an upper resolution. We choose here the modal value (mode) *i.e.*, the most frequently represented in $\mathcal{C}^l$. If $|\omega_w|$ denotes the cardinal of this class, then $\sum_{w=1}^{W} |\omega_w^l| = |\mathcal{C}^l| = 8$. We choose for the modelet:

$$p^{l-1} = \arg\max\{|\omega_w^l|, \omega_w^l \in \Omega_l\}\,.$$

It may happen that the above definition does not yield a unique maximum. If two or more categories dominate a cell block, a generic approach consists in taking into account its first block cell neighborhood (the surrounding 26 cells, except at mesh borders and boundaries). We affect the dominant value in the first neighborhood to p^{l-1}. In case of a draw again, the second-order surrounding can be used, iteratively. In practice for the presented version of HexaShrink, we limit to the first-order neighborhood, and choose the lowest indexed category when the maximum is not unique. Equipped with this unique lower resolution representative value, we proceed similarly to Subsection 3.2.2 for details, by using differences between original categories and the mode. As classes are often indexed by positive integers, a slight motivation allows to get only non-negative indices. By avoiding negative values, one expects a decrease in data entropy of around 5 %, which benefits to compression.

We thus change the sign of a detail coefficient if and only if it generates a value out of the range of $\{\omega_1, \omega_2, \ldots, \omega_W\}$, and then control this condition during reconstruction. So, all details $\{d_n^{l-1}\}$ for a cell block $\mathcal{C}$ are determined by:

$$d_n^{l-1} = (-1)^{(p_n^l - p^{l-1} < 0) \wedge ((2p^{l-1} - p_n^l) \notin \Omega)} \times (p_n^l - p^{l-1}).$$

During synthesis, the coefficients $\{p_i^l\}$ are obtained thanks to the closed-form equation:

$$p_n^l = p^{l-1} + (-1)^{\left((p^{l-1} + d_n^{l-1}) \notin \Omega\right)} \times d_n^{l-1}.$$

HexaShrink performs a decomposition of the mesh into a pyramidal structure, the inverse process perfectly reconstructs the data from the lower scale to the initial mesh, integrating details progressively. Adapted transforms are distinctly implemented on the various components, dealing with their dimensions and features as previously explained. The tool has firstly been designed for a visualization purpose, using thus multimedia concepts. Later, positive compression results demonstrated the HexaShrink capability as a compression tool. The decomposition generates sparser data. Then, encoding steps permit to reduce the data quantity by reordering the data, to a more compact storage size. The prime contribution consists in analyzing the process and experimenting the method on various meshes, in order to validate an appropriate generic approach.

3.3 Visual results: decompositions obtained with HexaShrink

Geosciences offer a huge variety of meshes, because of their various dimensions, structures and properties. We propose to validate HexaShrink on a representative benchmark of eight meshes. By this way we expect to experiment a generic methodology, suitable for various meshes.

The geological meshes are produced by petroleum companies during expensive drilling projects and seismic programs. These meshes are realistic, and carefully treasured by companies. Collecting geological meshes thus is a complicated quest, which explains the limited number of meshes in our benchmark. However, this dataset aims to be representative of the variety of meshes found in geosciences.

Our eight meshes are illustrated by Figure 3.11, and their main features are indicated in Table 3.1. They have various geometries (from smooth to fractured), and different dimensions (from small to large). To have an order of magnitude, the largest mesh (mesh#8) contains almost 380 times more cells than the smallest one (mesh#3). Also, several meshes contain continuous and/or categorical properties.

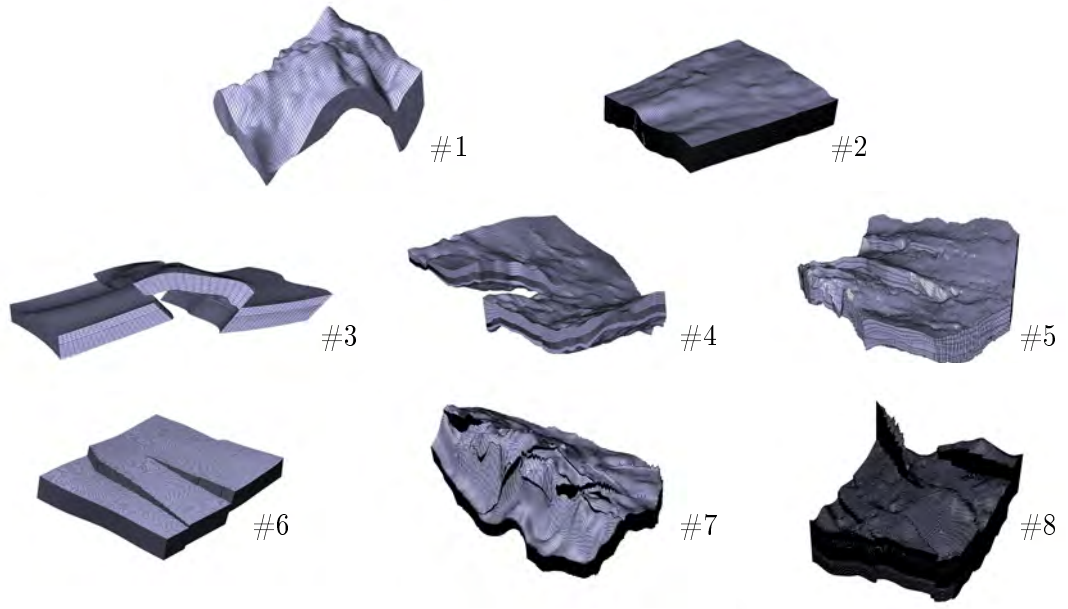


Figure 3.11: Our benchmark composed by eight meshes.

Figures 3.12 and 3.13 show respectively the decompositions of mesh#1 and mesh#8 provided by HexaShrink. The different resolutions are arranged in rows, by decreasing scale. The first column represents the mesh without any attribute. The second and the third columns represent the same mesh onto which a continuous and a categorical property is mapped, respectively.

Mesh#1 is decomposed to the lowest possible resolution (Figure 3.12, bottom), which is not very relevant from a geologist perspective. However, while all properties are almost constant, the lower arch corresponding to an anticlinal on the mesh at original resolution remains perceptible on the final "Lego brick" resolution. Looking at the porosity property (middle column), one observes how the values are progressively homogenized on coarser hexes. Concerning the rock type (last column), one observes that the modelet scheme tends to locally maintain predominant categories resolution after resolution, which is very satisfactory.

Figure 3.13 shows our largest mesh, mesh#8, that contains an isolated fault on the left side (the

Mesh index	Characteristics				Properties		
	# Cells	Dimension	Faults	File size	ACTNUM	Continuous	Categorical
1	93,600	$80 \times 45 \times 26$	No	4.62 MB	100 %	Porosity	Rock type
2	1,000,000	$100 \times 100 \times 100$	No	42.46 MB	100 %	—	—
3	36,816	$59 \times 39 \times 16$	Yes	1.46 MB	100 %	—	—
4	210,000	$100 \times 100 \times 21$	Yes	7.88 MB	20 %	—	—
5	450,576	$149 \times 189 \times 16$	Yes	22.73 MB	46 %	Porosity, Permeability	—
6	524,288	$128 \times 128 \times 32$	Yes	64.27 MB	100 %	Porosity, Permeability	
7	5,577,325	$227 \times 95 \times 305$	Yes	274.57 MB	97 %	Porosity	Rock type
8	13,947,600	$240 \times 295 \times 197$	Yes	580.94 MB	100 %	Porosity	Rock type

Table 3.1: Our collection of geological meshes with their ontological characteristics and geological properties.

diagonal crest shape) and a faulty block on the right. Even at the coarsest level, corresponding to a downsampling by $2^4 \times 2^4 \times 2^4$, these two structural discontinuities are still present, while keeping a good shape fidelity, globally. Concerning the attributes, the decompositions are also adequate.

Figure 3.14 confronts meshes mesh#5 and mesh#7 downsampled at power-of-two resolutions with HEXASHRINK and with the geomodeller SKUA-GOCAD™. SKUA-GOCAD™, but also PETREL™, are two geomodellers frequently used in geosciences to handle geological objects and to generate meshes for flow simulation. They include in particular upscaling/upgridding functions. We recall that upscaling/upgridding functions convert the properties and the geometry of a given mesh in a non-reversible manner to obtain a simplified version for flow simulation. Such functions are usually flexible yet often *ad-hoc*. We can see that HEXASHRINK tends to better preserve faults (colored in red). Figures emphasize an improved preservation of mesh borders, with an efficient management of ACTNUM throughout resolutions. Some artifacts may appear with SKUA-GOCAD™'s upgridding, which are automatically averted by HEXASHRINK, leading to nicer meshes at low resolution. As a summary, HEXASHRINK, while being fully reversible at dyadic scales only, efficiently and automatically manages structural discontinuities in the VMs. It may provide an interesting complement to existing irreversible upscaling/upgridding proposed by several geomodellers.

3.4 Conservative compression workflow for GMs

We just showed that HEXASHRINK is a powerful tool for structuring the GVMs into in pyramidal models, emphasizing the continuous zones, and rendering them sparser by transformation. It permits to display nice simplified versions, faithful to the original data. Additionally, HEXASHRINK should be also suitable for compression. Therefore, we will now study its potential compression performance, by combining it with an appropriate encoder. Our first experimentation is to combine HEXASHRINK with generic lossless encoders, to propose a conservative compression workflow.

We retain three generic lossless encoders: *gzip* (1992), *bzip2* (1996), and *LZMA* (1998). They are used by a general audience to compress diversified data. gzip has been proposed by Adler and Gailly. The first version was released in 1992 while optimized versions are still updated on zlib library. Its

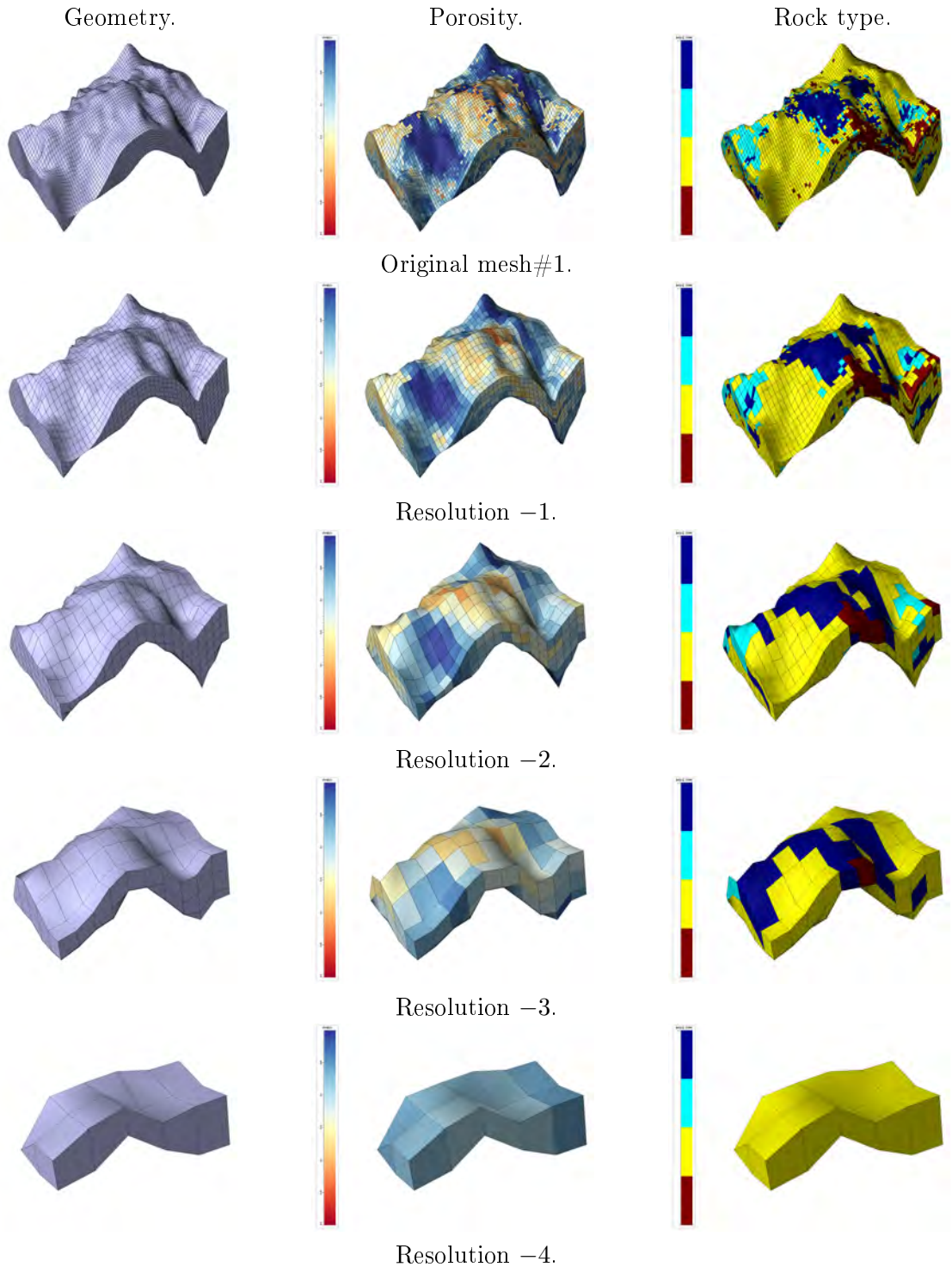


Figure 3.12: Original mesh#1, its attributes, and four levels of resolution generated with HexaShrink.

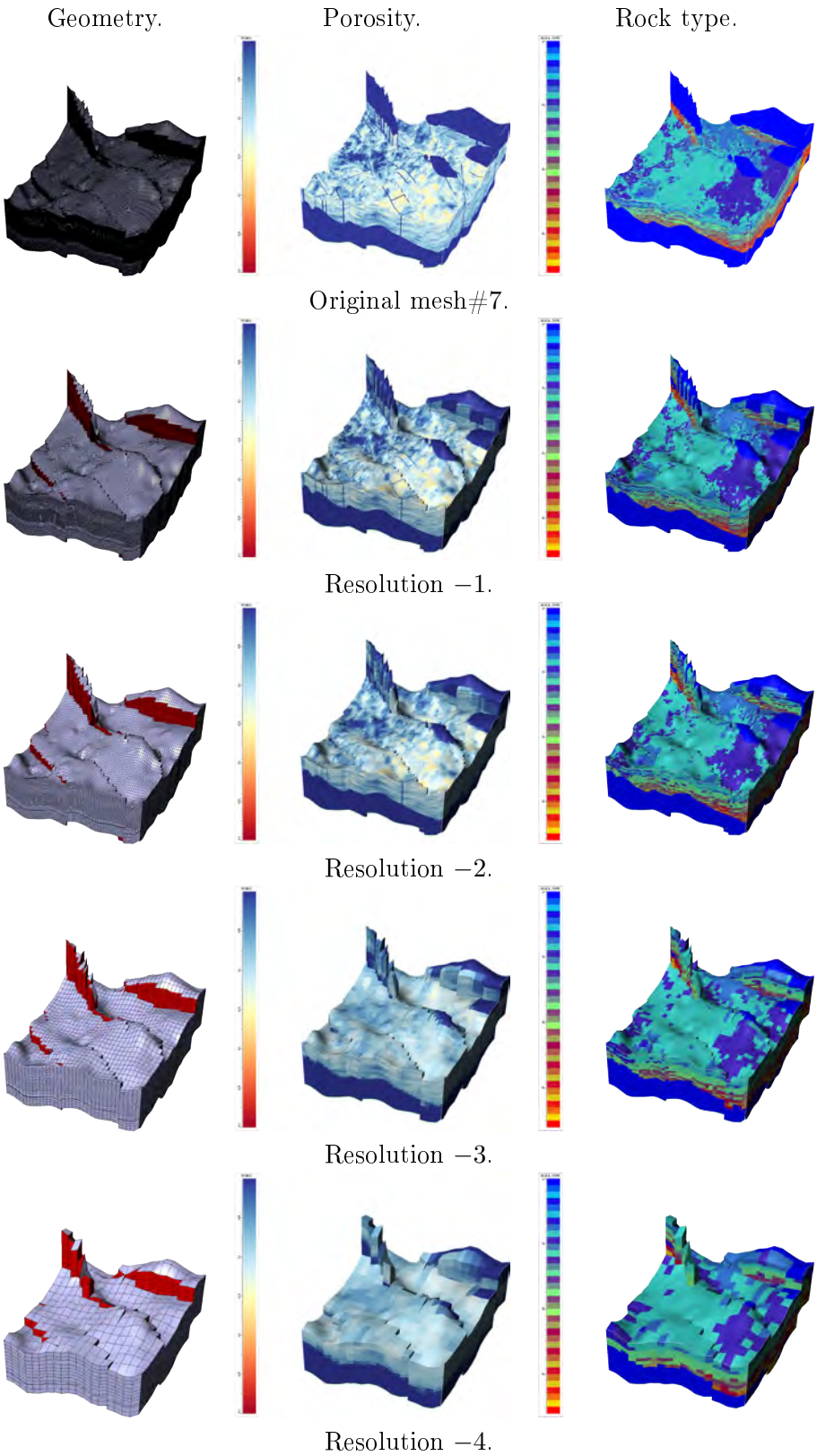
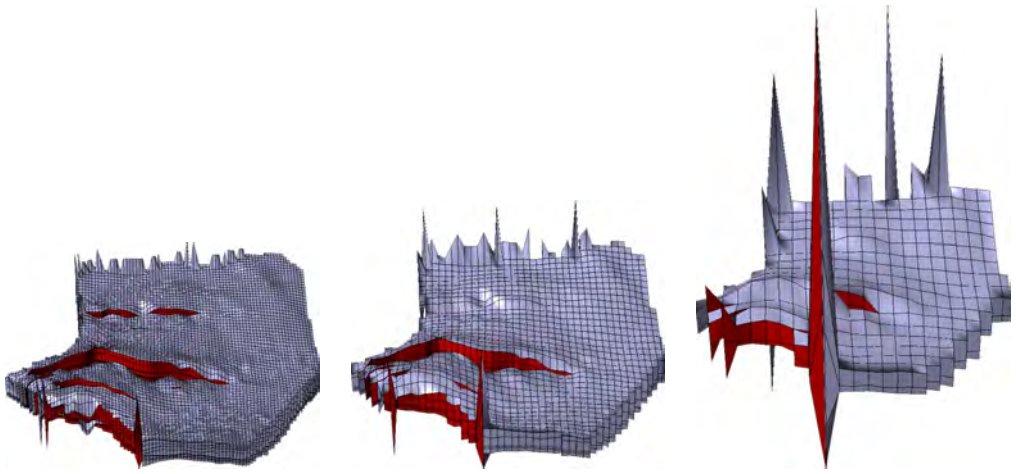
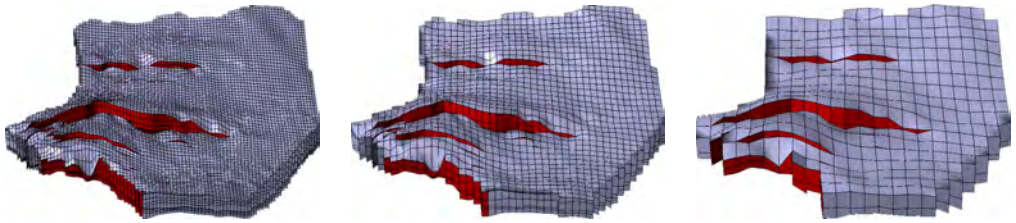


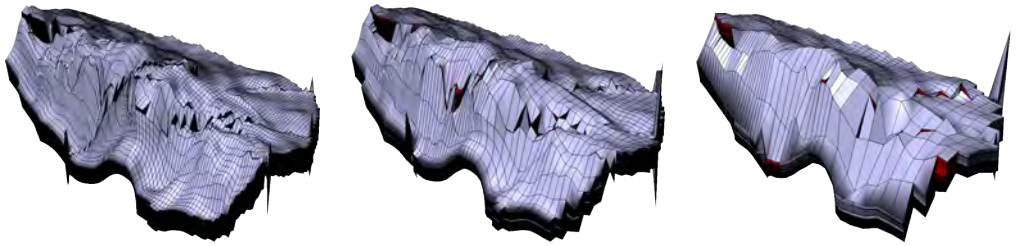
Figure 3.13: Original mesh#7, its attributes, and four levels of resolution generated with HEXA-SHRINK.



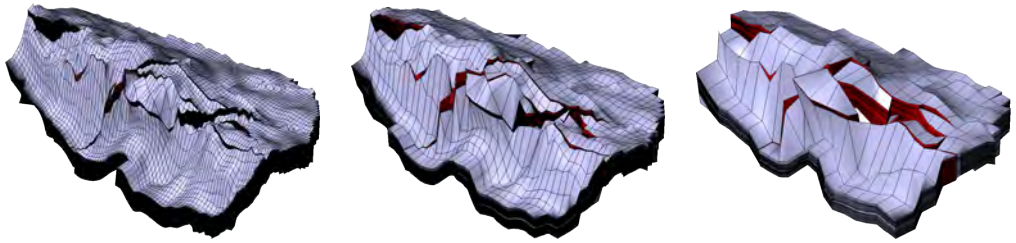
Mesh#5 with SKUA-GOCAD



Mesh#5 with HexaShrink



Mesh#7 with SKUA-GOCAD



Mesh#7 with HexaShrink

Figure 3.14: After dyadic downsampling/upgridding, HexaShrink (bottom) better preserves faults, and manages non-active cells (*i.e.*, with null Actnum values) across scales, yielding nicer borders at each resolution, contrary to GOCAD, as *examplified* with mesh#5 and mesh#7. From left to right: resolution -1, -2, and -3, respectively.

structure is based on a combination of LZ77 and Huffman algorithms. bzip2, developed four years later, is in direct competition with gzip, and uses the Burrows-Wheeler transform (Burrows and Wheeler, 1994), taking advantage of recurring patterns, and finishing also by an Huffman algorithm. The method yields better compression ratios with slower speed performance. Again four years later in 1998 LZMA (Lempel-Ziv-Markov chain algorithm) makes good compression performance, even better than both concurrents (gzip, bzip2) using a dictionary compression process. Its memory cost as its execution time are however higher. Such preliminary tools are not dedicated to scientific data unlike the following, but they illustrates the variety of approaches, and the complexity to develop an ideal tool for all applications and needs.

3.4.1 Coding performance

Table 3.2 presents the compression ratios obtained with gzip, bzip2, and LZMA applied to the outputs of HexaShrink. Each mesh component is encoded independently. A compression ratio (CR) is the ratio between the size of the raw data and the size of the compressed/encoded data. The higher the compression ratio, the more efficient the compression/encoder. The lines *none* indicate the CRs obtained when the raw data are directly encoded (HexaShrink is disabled). The lines with a number i indicate the CRs obtained when the mesh components are decomposed i times with HexaShrink, and then encoded. The main objective is to confirm the contribution of HexaShrink in the size reduction whatever the encoder used, but also to determine the most efficient for a further implementation on HexaShrink.

We focus our analysis on the mesh#5, a quite complicated and faulty model. Without decomposition, we observe variable CRs according to the coder: from 2.46 for gzip to 3.33 for LZMA. The similar observation is made for the other meshes, except for the mesh#7, in which gzip is slightly more efficient than bzip2. But whatever the mesh, LZMA provides the best compression results. With only one level of decomposition, we can see a significant improvement. Always considering the mesh#5, the CR obtained with LZMA increases up to 3.71. A growing gain up to 3.81 is obtained using a second decomposition, yet additional decompositions only provide a marginal improvement.

The HexaShrink decomposition improves the efficiency of the coders. This observation could be generalized on the global Benchmark, yet LZMA performance decreased on the mesh#8 using HexaShrink. This might be explained by certain mesh modeling practice with a geomodeller: horizons of the formation layers are preliminary modeled, before incorporating it into the mesh. The user first determines for each layer the number of cells in height, and then the thickness of the layers are computed by interpolation. Hence, locally, an observed value v may arise from a scale s and an offset o relationship $v = s \times m + o$, on some integer index m, easier to compress than v. LZMA's superior capability owes to its capacity to capture complex models of byte patterns. By contrast, with a wavelet decomposition, the affine relationship is poorly captured throughout approximations, due to the rounding in wavelet lifting (*cf.* Subsection 3.2.1). Hence, multiscale decompositions may slightly reduce the raw compression performance for meshes presenting initial "numerical format" artifacts, or illusory floating-point precision. This however does not hamper the usability of HexaShrink for storage and visualization, as the direct access to a hierarchy of resolutions respecting discontinuities is granted, while already providing impressive compression rates of about 8–9, superior to most results for the others meshes.

Mesh	Level	gzip	bzip2	LZMA
1	none	3.73	4.98	6.43
	1	5.62	6.07	7.52
	2–4	5.67	6.12–6.13	7.42–7.44
2	none	3.23	8.41	10.12
	1	6.49	10.82	11.81
	2–6	7.48–7.58	12.75–13.03	13.35
3	none	2.67	2.99	3.63
	1	3.88	4.70	5.24
	2–4	4.03–4.05	4.92–4.93	5.47–5.48
4	none	1.83	1.89	2.21
	1	2.64	3.06	3.48
	2–4	2.76	3.22–3.23	3.64–3.65
5	none	2.46	2.55	3.33
	1	3.14	2.83	3.71
	2–4	3.25–3.26	2.91–2.92	3.80–3.81
6	none	1.88	2.25	3.04
	1	2.70	3.17	3.71
	2–6	2.84–2.86	3.39–3.42	3.90–3.93
7	none	2.32	2.25	3.04
	1	3.31	3.53	4.44
	2–6	4.14–4.24	4.48–4.68	5.54–5.73
8	none	3.20	5.98	12.52
	1	5.42	7.07	8.90
	2–7	5.80–6.72	7.63–10.12	9.05–10.23

Table 3.2: Coding performance of our conservative compression workflow. HexaShrink is combined with gzip, bzip2 and LZMA, for different levels of decomposition.

3.4.2 Speed performance

The execution time is also a significant issue, a delay could indeed become critical depending on the application. In the present situation, we may resort to asymmetrical compression-decompression schemes, sometimes termed "compress once, decompress many". The encoding can take more time because it is performed only once, to optimize storage and transfer. Rather, decoding should be faster to be performed several times, on demand. The speed performance of the conservative compression workflow are highly dependent on the mesh, its complexity, its dimensions and the level of decomposition. For a baseline evaluation, a Java implementation was run on a laptop with Intel Core i7-6820HQ CPU @ 2.70 GHz processor and 16 GB RAM. Each mesh was compressed to the maximum level, and decompressed, twelve times. As the outcomes were relatively stable, they were averaged.

Figure 3.15 presents the execution times of our conservative compression workflow in function

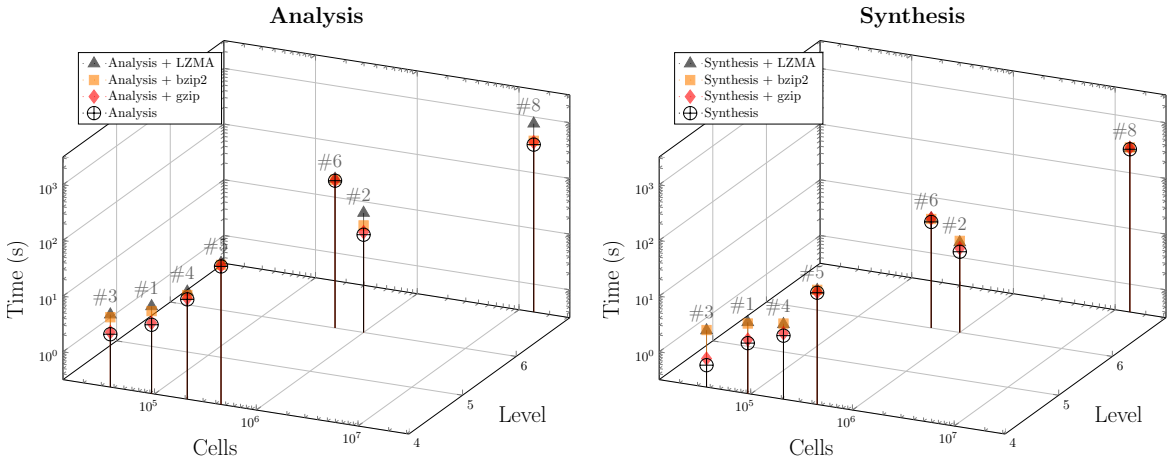


Figure 3.15: Comparison of the execution times of our conservative compression workflow depending on the encoder. Timings for analysis and synthesis are also presented.

of the encoders used. Timings of the analysis and synthesis are also presented. The synthesis stage (i.e., the inverse transform of HexaShrink) is faster than the analysis, up to 4 times for instance for small meshes such as mesh#4, while this difference reduces with larger meshes. Of the three generic coders, gzip is definitely the fastest in general for coding and decoding, followed by bzip2 and LZMA. LZMA necessitates more times to code the data, but is faster than the bzip2 to decode the largest meshes as mesh#8. This observation supports our previous expectation, promoting a fast decompression to be performed several times against a slower compression, performed only once. Moreover, the LZMA speed performance confirms its efficiency already demonstrated with the compression rates.

3.4.3 In-depth analysis of the coding performance

As previously explained, a GM is a composite object made of a geometrical structure and continuous, categorical properties. The components are separately treated by an adapted wavelet, generating for each component a hierarchical structure.

To complement the global coding performance of the conservative compression workflow presented in Subsection 3.4.1, we now study and compare the performance achieved for each mesh component (Zcorn, Pillar, Activity, Continuous and Categorical properties) to demonstrate the relative efficiency of the various transforms integrated in HexaShrink. The histogram of Figure 3.16 presents the average binary cost (number of bits per value or symbol) for each component of the mesh#5 with LZMA, in function of the number of successive HexaShrink decompositions. As a reference, the bar named *Raw* gives the native binary cost of each mesh component. The results for the entire benchmark are available in appendices (page 119).

According to the component and its required precision, the native values are written on 64 bits,

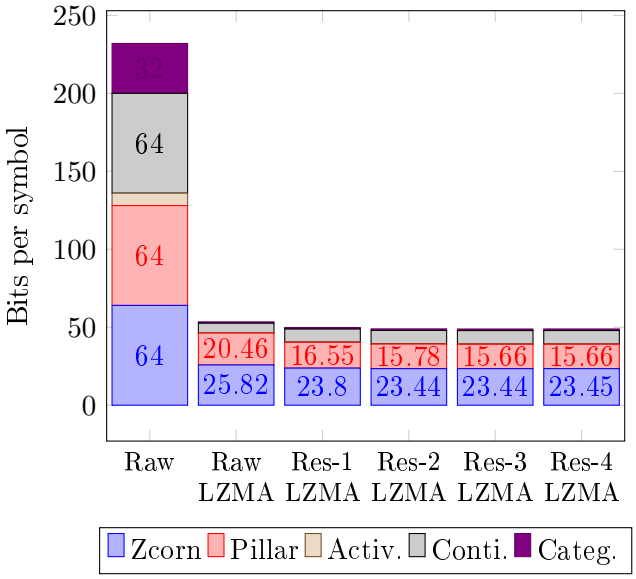


Figure 3.16: Binary cost of each component on mesh#5 in function of the number of successive HexaShrink decompositions.

32 bits or 8 bits, corresponding to standard formats. Applied on the raw data, LZMA efficiently reduces the number of bits whatever the considered component: an altitude value (Zcorn), coded on 64 bits only needs 25.8 bits through LZMA. One first HexaShrink decomposition allows reducing it by 2 bits in more, and a second by 0.4 bits. As previously shown with the global mesh results (the gains become modest after two decomposition levels). The same goes for the coordinates values (Pillar), both constitute indeed the geometry, and wavelet methods have already proven to be very efficient on this kind of data.

Meanwhile, the decomposition applied on the continuous property does not have the expected impact. LZMA significantly reduces the native 64 bits needed to encode a cell label (up to 6.28), but one HexaShrink decomposition slightly increases (up to 8.46). This limited effect could be explained by a high magnitude and an over numerical precision required by the simulation. Besides the lossless generic encoders do not efficiently use the inter-scale redundancy in the multiscale structure generated by HexaShrink. Such observations encourage us to test more evolved coders, such as the zerotree coders. This concept will be developed in the next subsection. Also, we observe that the transformation employed currently in HexaShrink for the cell activity is not the most appropriate. The cell activity consisting in binary values, we believe that a binary wavelet (Pigeon and Bengio, 1999) should be more performant. It will be tested in the future.

Finally, with this benchmark study, we verified the relevance of using HexaShrink in a compression workflow. The global performance, as the execution time, obtained on the global meshes are meaningful and promising for our future work. However, the in-depth analysis for each mesh component reveals a disparity according to the type of property, for the continuous ones in particular. More evolved encoders could provide an adapted solution for both, exploiting the correlation

inter subbands and offering performant compression at refinable precision.

3.5 Compression workflow at refinable precision for GMs

The HexaShrink methodology yields a multiscale representation of each mesh component. As illustrated by Figures 3.12 and 3.13, the renderings at different scales exhibit an evident visual correlation. This inter subband correlation can be employed to improve the coding performance, with the help of a coder at refinable binary precision.

3.5.1 How coding at refinable precision?

The notion of coding/decoding (or compression/decompression) at refinable precision is illustrated by Figure 3.17. In a nutshell, let us consider a 1D vector containing eight integer values (top-left), from zero to 15. Typically, the binary cost of each value is 4. Each original value can thus be progressively encoded, from the Most Significant Bit (MSB_O) to the Least Significant one (LSB_O) plus a sign bit, as shown in the left part of the Figure 3.17 (the index O refers to the original data). During the decoding stage, we can reconstruct the original vector at refinable precision by using only a limited number of *bit planes*, starting from the MSB_O (plus the sign bits). On the left-part of Figure 3.17, we can also see the vector reconstructed with only three bit planes (in that case, $\text{LSB}_O = 1$), and with only two bit planes (in that case, $\text{LSB}_O = 1$). The lower the number of bit planes, the less precise the reconstructed vector.

This principle can be also applied on transformed or decomposed data. In the right part of Figure 3.17, we show for example the original vector on which a simple |2|-level S+P transform (for sequential/prediction, *cf.* Said and Pearlman (1993),) has been applied to. This transform, that can be considered as a non-linear avatar of the Haar wavelet (Haar, 1910; Jacques et al., 2011), yields several subsets of values: one subset of approximation coefficients, and several subsets of details, two in our case. If $s_l[k]$ denotes a sequence of approximation coefficients at a given level l, the S+P transform yields two half-length subsequences:

$$s_{l+1}[k] = \left\lfloor \frac{s_l[2k] + s_l[2k+1]}{2} \right\rfloor , \tag{3.10}$$

$$d_{l+1}[k] = s_l[2k] - s_l[2k+1] , \tag{3.11}$$

with $d_l[k]$ denoting detail coefficients at level l.

The resulting multiscale data (shown at top-right, the red bars separating the different subsets) can be also fully encoded with 4 four bit planes, and also decoded partially with a limited number of bit planes, before reconstruction. In that case, we can also reconstruct at refinable precision.

That being said, some remarks can be done, to emphasize the relevance of using transformed data. The absolute values of the combined multiscale data are globally smaller in amplitude, as the transform captures data regularity at different scales. As a consequence, only one symbol "1" remains in the MSB_M, and for an approximation coefficient. At the opposite, the symbols "1" representing the details in the multiscale data are mainly concentrated in the LSB_M, which allows reducing

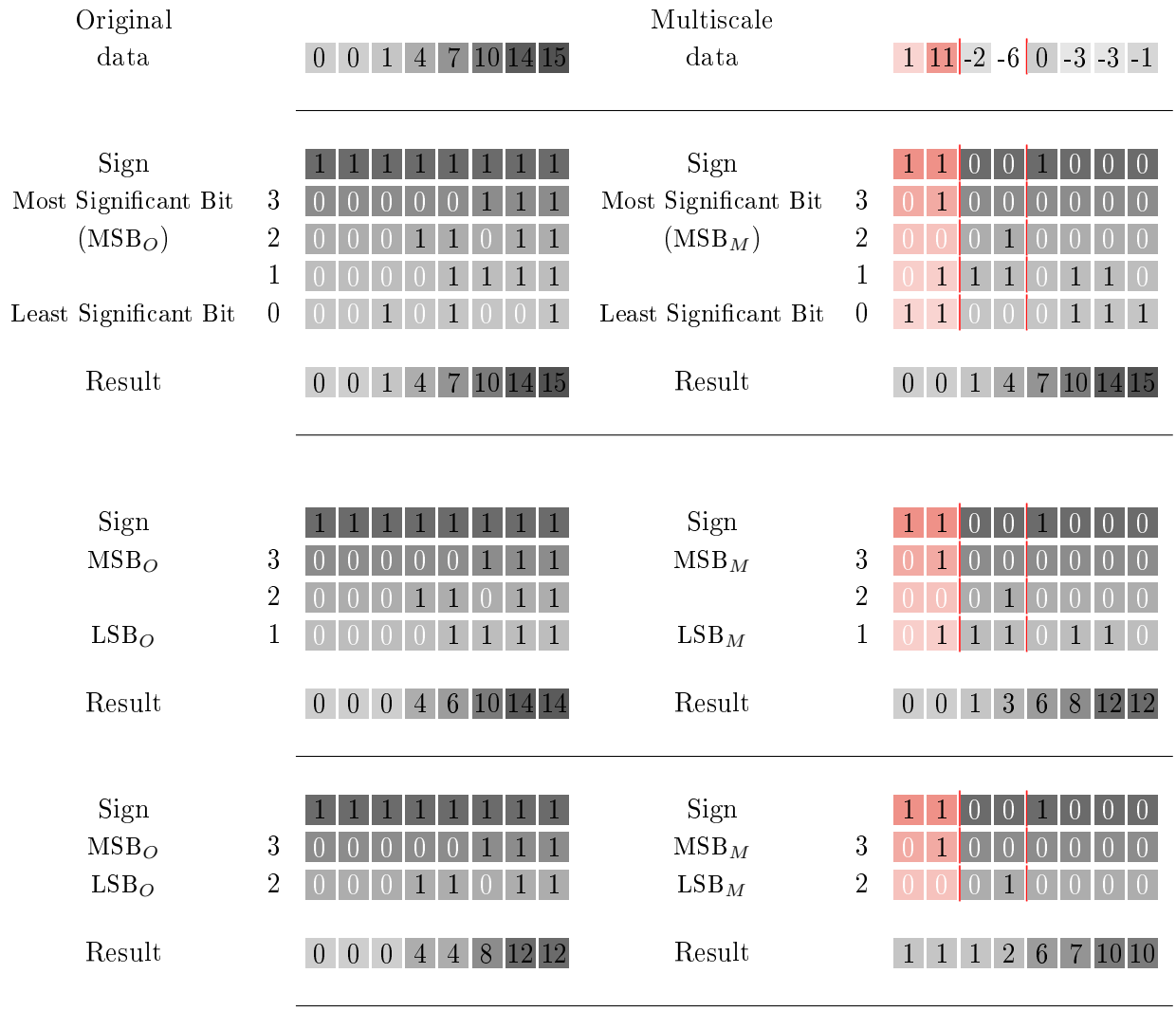


Figure 3.17: A textbook case with a 1D vector to show the interest of encoding transformed/decomposed data at refinable precision. (Left) the original vector coded/decoded at refinable precision. The index O refers to the original data. (Right) a multiscale version of the original vector (obtained by transformation) coded/decoded at refinable precision. The index M refers to the multiscale data.

the global binary cost. To illustrate this very interesting feature in a context of compression, let us simply calculate the total number of bits required to perfectly retrieve the original vector from the original bit planes and from the bit planes representing the multiscale data. By neglecting the sign bit, 21 and 16 bits are respectively necessary, which attests that a transformed/hierarchical data can facilitate the encoding. Moreover, the same phenomenon occurs when the reconstruction is done at refinable precision, as shown in the middle and bottom parts of Figure 3.17.

This methodology can be generalized to data of higher dimensions, using dependencies between the subsets of different levels. Since approximations shrink by a factor of two in each dimension (cf. Equation 3.10), 2^d blocks of values at level l relate to one value at level $l+1$. This results in binary trees, whose the inheritance between the subbands are depicted in Figure 3.18 for $2D$ and $3D$ data.

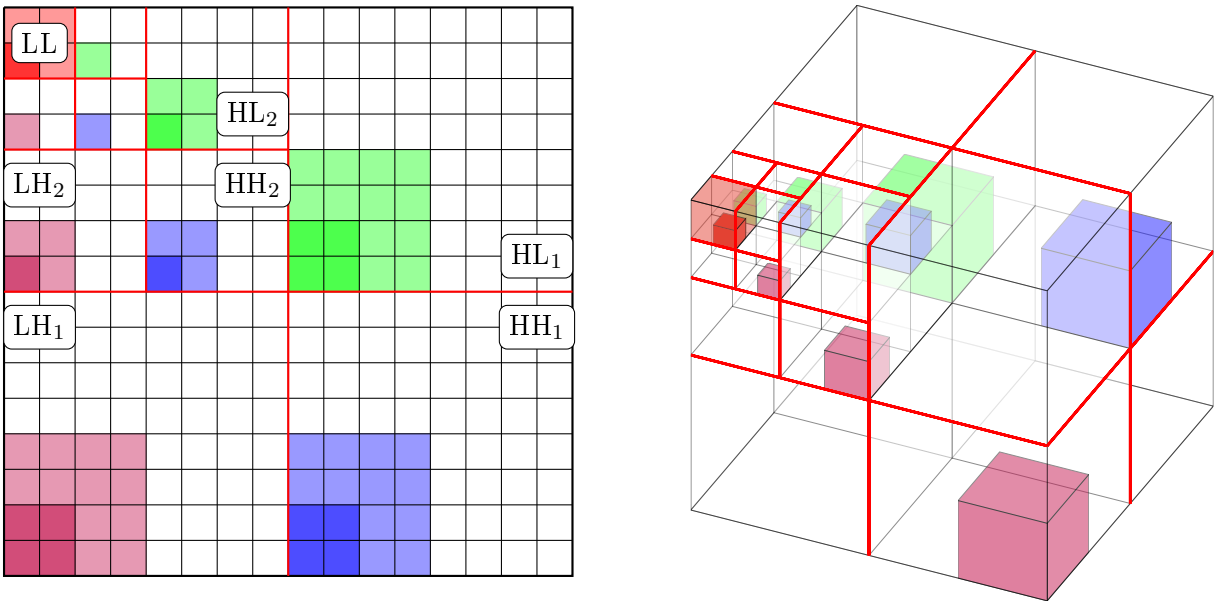


Figure 3.18: Inheritance between the subbands generated by wavelet-based decomposition on 2D and 3D grids.

3.5.2 Principle of zerotree coding

When multiscale data are parsed from most to least significant bits according to the binary trees, a significant quantity of 0 is very often observed. In that case, the binary trees are called *zerotrees* (ZT). They have been popularized for image compression, early with EZW (Embedded Zerotree Wavelet, (Shapiro, 1993)), later with SPIHT (Set Partionning In Hierarchical Trees or SPIHT, Said and Pearlman (1996)).

The principle of ZT coding is to replace a tree of zeros with a single symbol to reduce the information quantity. To succeed, the root node of the tree should be identified. It corresponds to the lowest scale, while related coefficients in higher frequency subbands draw the tree branches. After an appropriate wavelet decomposition, structured data become parsimonious, generating sub-

trees mostly composed of zero or close to zero coefficients. By property inheritance, an insignificant coefficient in a particular subband is likely to have insignificant descendants in related subbands. The principal mechanism for data coding consists in localizing significant root coefficients above a given threshold. Then bit plane per bit plane, descendant coefficients below the threshold are coded globally as zerotrees. The progressive compression process starts with a threshold close to the maximal coefficient. When a wavelet bit plane (corresponding to amplitudes between 2^{q-1} and $2^q - 1$) is encoded, the threshold is reduced to describe the lower bitplane. By first coding significant features, and progressively thereafter smaller details, this approach better takes into account the inherent wavelet decomposition structure than external lossless coders. The transmission of a ZT coded data could be interrupted at any time, resulting in a (de)compression at refinable precision. The truncated data would correspond to a more or less accurate approximation of the initial data. This is lossless/conservative when all bit planes are transmitted.

Combining a ZT coder with HexaShrink seems to be a promising way to encode GMs in a progressive way, to finally provide a compression/decompression workflow at refinable precision, and at refinable resolution. Aside pure compression gains, we are interested in the potential of ZT to contribute to simulation efficiency. Indeed, works are currently devoted to finding alternatives to floating-point representations (Muller et al., 2018; Lindstrom et al., 2018; Cappello et al., 2019). A ZT-based structure permits to decrease the grid resolution in a dyadic fashion, and to refine the precision by power-to-two factors. Its combination with multiscale approximations is illustrated by Figure 3.19: starting with properties values coded on 5 bits (top-left image), the grids can be represented at three different resolutions, and at refinable precision by limiting the number of significant bits ($n\text{MSB}_M$).

3.5.3 Coding performance

Implementing a ZT coder for 3D data is not trivial. An experimental version is implemented within the JPEG 2000 standard (Schelkens et al., 2006), and serves as a reference to several works in the field. During this study, we use the algorithm developed by Christophe et al. (2008) for $3D$ Cartesian hyperspectral satellite data. It was kindly communicated by the CNES, and we ponder on the opportunity of a collaboration, in a joint effort toward knowledge advancement. This tool was developed with the anisotropy of hyperspectral data in mind, a feature also observed in geosciences, along depth (*cf.* Subsection 4.3.1). Our code is also based on *QccPack* library, developed by Fowler (2000) in open source. Some adaptations have been required, considering precision and dynamics of our raw data.

Because the mesh#6 is complete and has ideal dimensions for the reservoir simulations presented in the next chapter, unlike other meshes[1], all our tests with the ZT coder will be performed on this mesh. In addition, we only present the CR obtained on continuous properties, as we showed in the previous section that these data are the most delicate to compress.

The continuous properties porosity and permeability of the mesh#6 are inspired by the tenth SPE

[1]Seven other meshes in the benchmark are sedimentary basins and have kilometric spatial dimensions. The average distance of influence between two wells is 300 m, so the dimensions of the basins are much too large to consider simple cases.

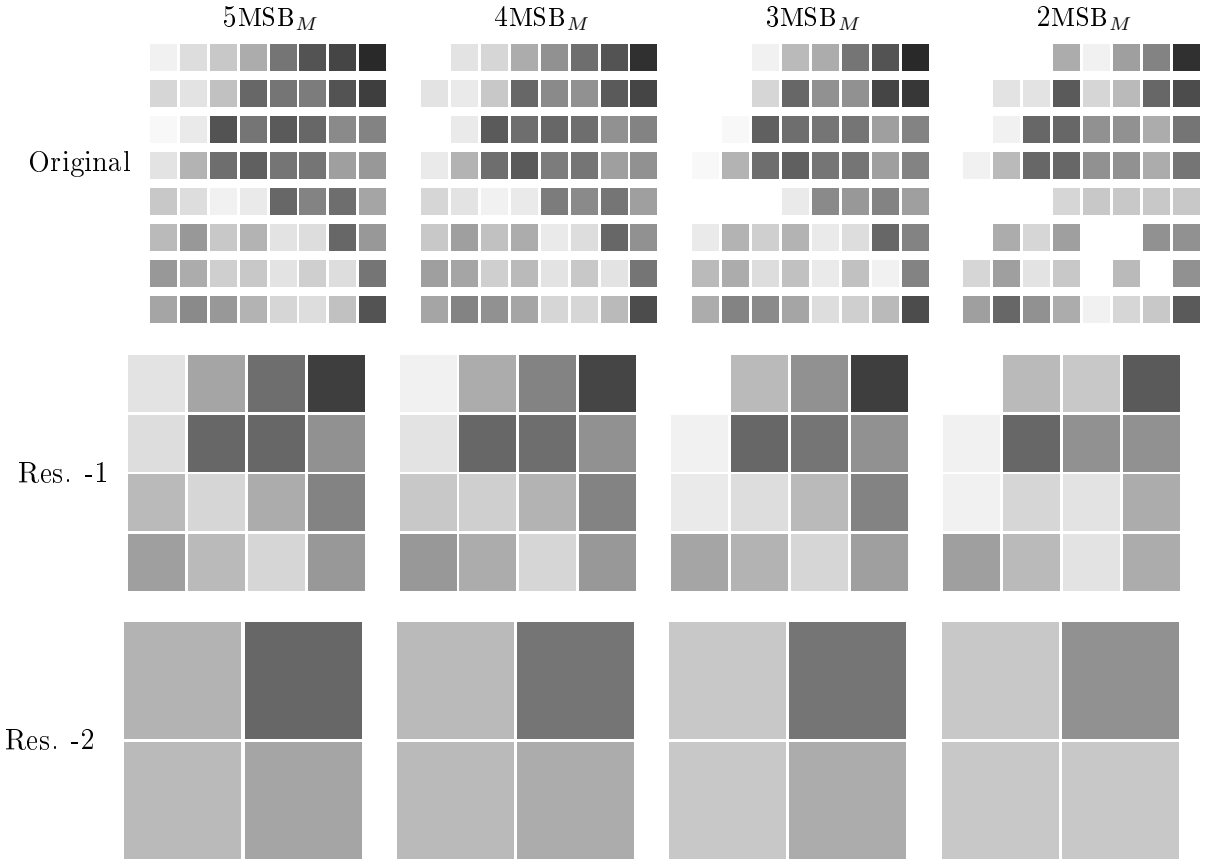


Figure 3.19: 2D data at three different resolutions (Haar wavelet) and refined numerical precision: from the original with 8×8 values in 0–31 with 5MSB$_M$ (top-left) to a two-fold coarser resolution coded with 2MSB$_M$ (bottom-right).

Comparative Solution Project (also known as SPE10) Christie and Blunt (2001). We experimented two formations (cf. Figure 3.20): a *Tarbert* formation, prograding *near shore environments*, and a *Uperness* formation, prograding *fluvial environments*.

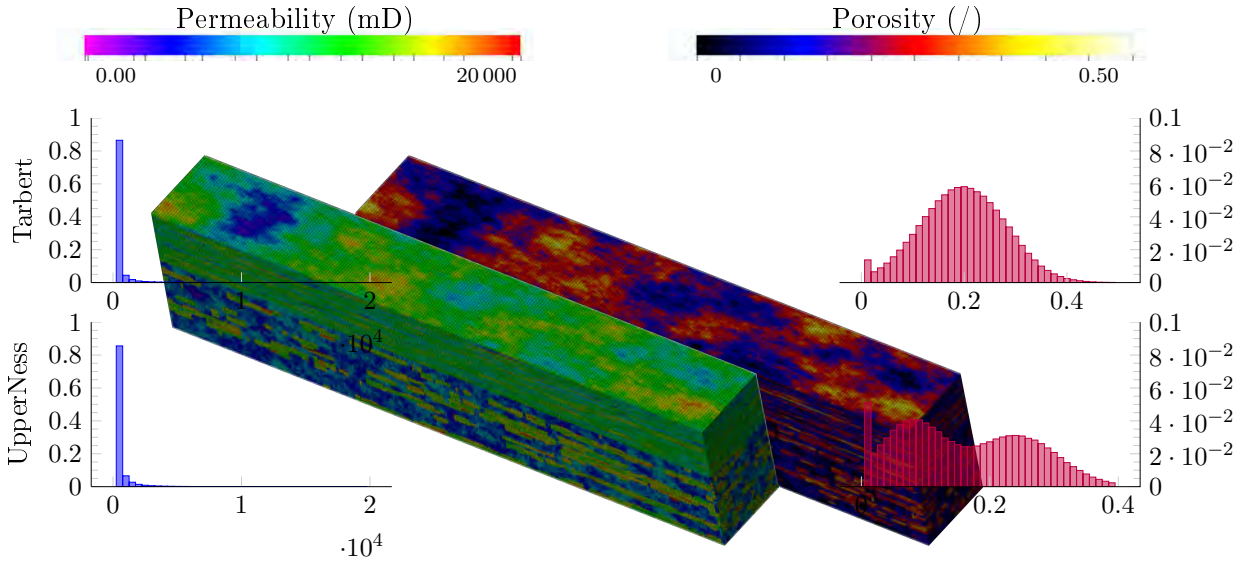


Figure 3.20: Visualization of the permeability (left) and porosity (right) properties of SPE10. The four histograms show the distribution of these properties according two environments: *near shore* environment (Tarbert formation on top) and *fluvial* environment (Upernees formation on bottom).

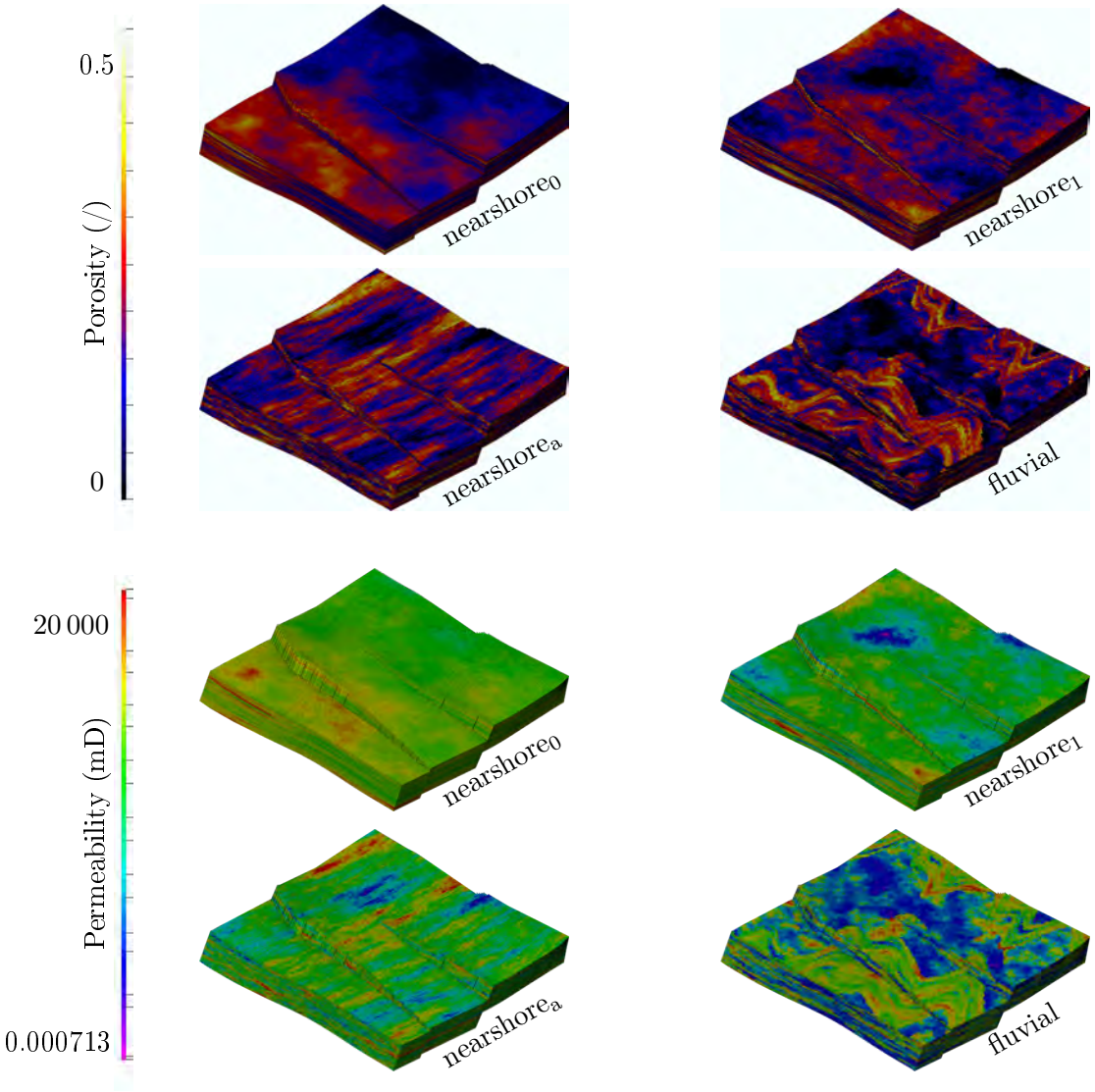


Figure 3.21: Visualization of the eight sets of continuous properties tested with the ZT coder on the mesh#6, generated from four distinct geological environments. The dynamic range of the distribution is 0 to 0.5 for porosity, and 0.0007 to 20 000 mD for permeability, except with nearshore_0, whose the minimal permeability value is higher (4.7 mD).

The Tarbert formation inspired us three models, named hereinafter nearshore_0, nearshore_1 and $\text{nearshore}_\text{a}$ (the index "a" indicates an anisotropic distribution along one direction). The Tarbert formation conditioned one model, named hereinafter fluvial. The four resulting datasets are shown in Figure 3.21. They mainly differ in their spatial distribution: the three nearshore environments present smooth variations, while the fluvial environment exhibits sharper contrasts, with distinctive heterogeneous geological objects. This discrepancy between environments primarily allows a wide range of petrophysical behaviours.

Our first experimentation is to encode the eight datasets presented just above with ZT in lossless mode, and to compare its CRs with those obtained with LZMA in the same context. Table 3.3 shows the CRs obtained with four combinations of transforms and encoders:

- HS_P+LZMA: the transform for continuous properties proposed in HexaShrink (presented in Subsection 3.2.2, page 47) followed by LZMA;
- HS_P+ZT: the same transform followed by ZT in lossless mode;
- Haar+ZT: the transform HS_P is substituted by the classical 3D Haar transform;
- CDF 9/7+ZT: the transform HS_P is substituted by the popular CDF 9/7 transform.

Regarding the properties of the nearshore_0 environment, the best CR is obtained when LZMA is applied directly to the data, without any transform (line *none*). One level of HexaShrink decomposition decreases the LZMA efficiency from 2.79 to 2.12, and to 2.11 for several levels of decomposition. By substituting LZMA with ZT, we expected an improvement, but finally obtain a poorer result, at 1.93. This fact reveals that the transform proposed in HexaShrink for the continuous properties is not optimal in term of compression performance. The reason is that this transform considerably increases the dynamic range of decomposed data, in order to ensure a fully reversible decomposition. Haar and CDF 9/7 better perform, but remains lower than LZMA after all. Our explanation bases on the environment physiognomy: smoother and more homogeneous. Therefore the performance of an evolved encoder using the residual dependencies within multi-scale decomposition are less remarkable. However, regarding the three other environments, the classical wavelets Haar and CDF 9/7 combined with ZT outperform HS+ZT again, itself outperforming LZMA alone. Similar results are obtained for the permeability.

We can thus conclude that, globally, ZT seems to be a better solution than generic lossless encoders. Also, the transform proposed in HexaShrink seems to be less appropriate than Haar or CDF 9/7 to get the best compression performance on the continuous properties. On the other hand, it ensures a fully reversible decomposition, contrary to Haar or CDF 9/7. Indeed ZT needs integer values as input, a quantization before using it is thus required. Nevertheless, we will see hereinafter that this slight "degradation" does not necessarily have an impact on the visual quality and on the simulation results, which allows us to consider a ZT-based workflow as "quasi-conservative".

Our second experimentation is to use the ZT coder in progressive mode, in order to evaluate its efficiency in the context of compression at refinable precision. By progressively decoding the bit planes, from the MSB to the LSB, a SNR curve can be drawn in function of the quantity of bits decoded, and thus for different precisions. Figure 3.22 shows the resulting SNR curves when the Haar transform is used. Each graph analyzes one of our eight continuous properties presented before. Each marker corresponds to a specific number of bit planes decoded. The rightmost marker of each curve corresponds to the CR obtained in lossless mode (Haar+ZT in Table 3.3), i.e., when all the bit planes are used. For purposes of comparison, the bit budgets corresponding to the CR obtained for the methods LZMA, HS_P+LZMA and HS_P+ZT in lossless mode (previously presented in Table 3.3) are also represented in each graph (by three vertical straight lines).

Porosity	Level	HS_P LZMA	HS_P ZT	Haar ZT	CDF 9/7 ZT
nearshore0	none	2.79			
	1	2.12			
	2–5	2.11	1.93	2.36	2.40
nearshore1	none	1.69			
	1	2.10			
	2–5	2.09	1.93	2.29	2.30
nearshore ani	none	1.68			
	1	2.10			
	2–5	2.07	1.94	2.26	2.28
fluvial	none	1.60			
	1	2.11			
	2–5	2.09	1.93	2.18	2.20

Permeability	Level	HS_P LZMA	HS_P ZT	Haar ZT	CDF 9/7 ZT
nearshore0	none	2.76			
	1	1.55			
	2–5	1.53	1.34	2.17	2.18
nearshore1	none	1.80			
	1	1.53			
	2–5	1.52	1.34	2.10	2.07
nearshore ani	none	1.78			
	1	1.51			
	2–5	1.51	1.34	2.05	2.03
fluvial	none	1.84			
	1	1.55			
	2–5	1.54	1.34	2.02	1.97

Table 3.3: Comparative (near) lossless coding performances on porosity (top tabular) and permeability (bottom tabular) property from our four environments with compression ratios at different decomposition levels combining HexaShrink with LZMA, zerotree, and Haar wavelet transform with ZT.

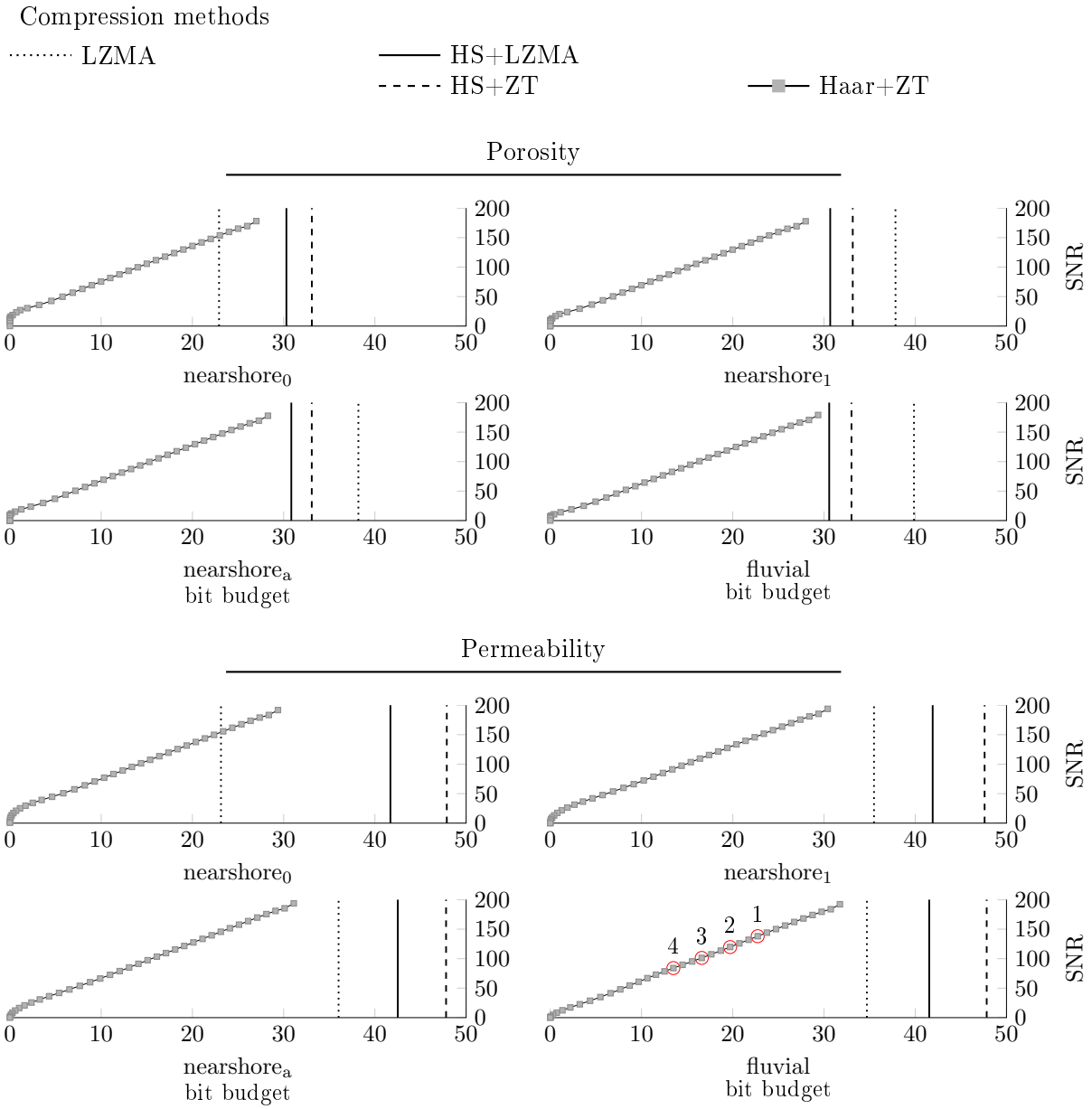


Figure 3.22: SNR curves of four compression workflows. Three of them are conservative (LZMA, HS_P+LZMA, HS_P+ZT in lossless mode), while Haar+ZT is used in progressive mode. The marks are obtained thanks to a progressive decoding of the bit planes. The four red circles in the bottom-right graphic refer to visual representations of Figure 3.23.

As expected, a (de)compression workflow at refinable precision allows reaching bit budgets much more smaller than the lossless methods. In the same time, the SNR inevitably decreases according to the bit budget, indicating data less and less precise, and thus less and less faithful to the initial data.

A trained eye would notice the shape of our SNR curves differs from the SNR curves classically obtained for images and videos (whose pixel value is commonly encode on less bits). Usually, the curves increase sharply to gradually reach a plateau at a certain bit budget. At this point progressive decompression can stop, because a higher binary budget would not significantly improve the objective quality. Considering our results obtained on scientific data, the interpretation is more difficult because curves are incredibly linear. The same trend is noticeable in other scientific studies working on similar topics (Cappello et al., 2019). This problem could be addressed by testing other metrics.

To have an idea of the impact of a (de)compression at refinable precision in a context of geological data visualization, Figures 3.23 and 3.24 show the renderings of two given permeability properties decompressed at different bit budgets. For comparison purposes, the original data are also shown on the top corner left of each figure. On the first figure we can oberve that the visual degradations are quasi imperceptible until a bit budget of 19.71 bits/value, which corresponds to a much lower bit budget than those obtained with the lossless methods. A similar observation can be done on the second figure until a bit budget of 18.85 bits/value. Moreover, as a teaser of the compandor effect on continuous properties (see next chapter), these figures also show the high quality of the renderings obtained for these two properties with only 12.65 and 12.17 bits/value respectively (bottom corner right of each figure), when a compandor is applied before the compression workflow. The visual results are even more satisfactory.

All these results are very promising, and validate the fact that processing the continuous properties of geological meshes with a compression workflow at refinable precision does not necessarily affect the data severely for certain applications. This is the case for visualization, but we will see in the next chapter that the same observations can be made in the context of simulation.

Conclusion of this chapter In this chapter we have shown that a suitable multiscale representation such as HexaShrink can improve the compression workflow for the storage and the visualization at different levels of detail of geological meshes. HexaShrink is the result of a literature review of existing approaches dealing with VMs, in particular geological, and other more general scientific data. This solution gathers different wavelet-based transforms adapted to the heterogeneous components of the GMs. Firstly developed for visualization (Peyrot et al., 2019), it decomposes an initial mesh into embedded hierarchical structure. The process is fully reversible and generates lower resolutions while preserving the mesh features across resolutions (as faults and properties coherency).

My first contribution consisted in validating the skills of HexaShrink in a comparative study with upgridding techniques available in popular geomodellers, on a representative benchmark composed of eight meshes. Then, we used HexaShrink in the context of compression, by combining it with generic encoders. We observed that globally HexaShrink improves the coding performance,

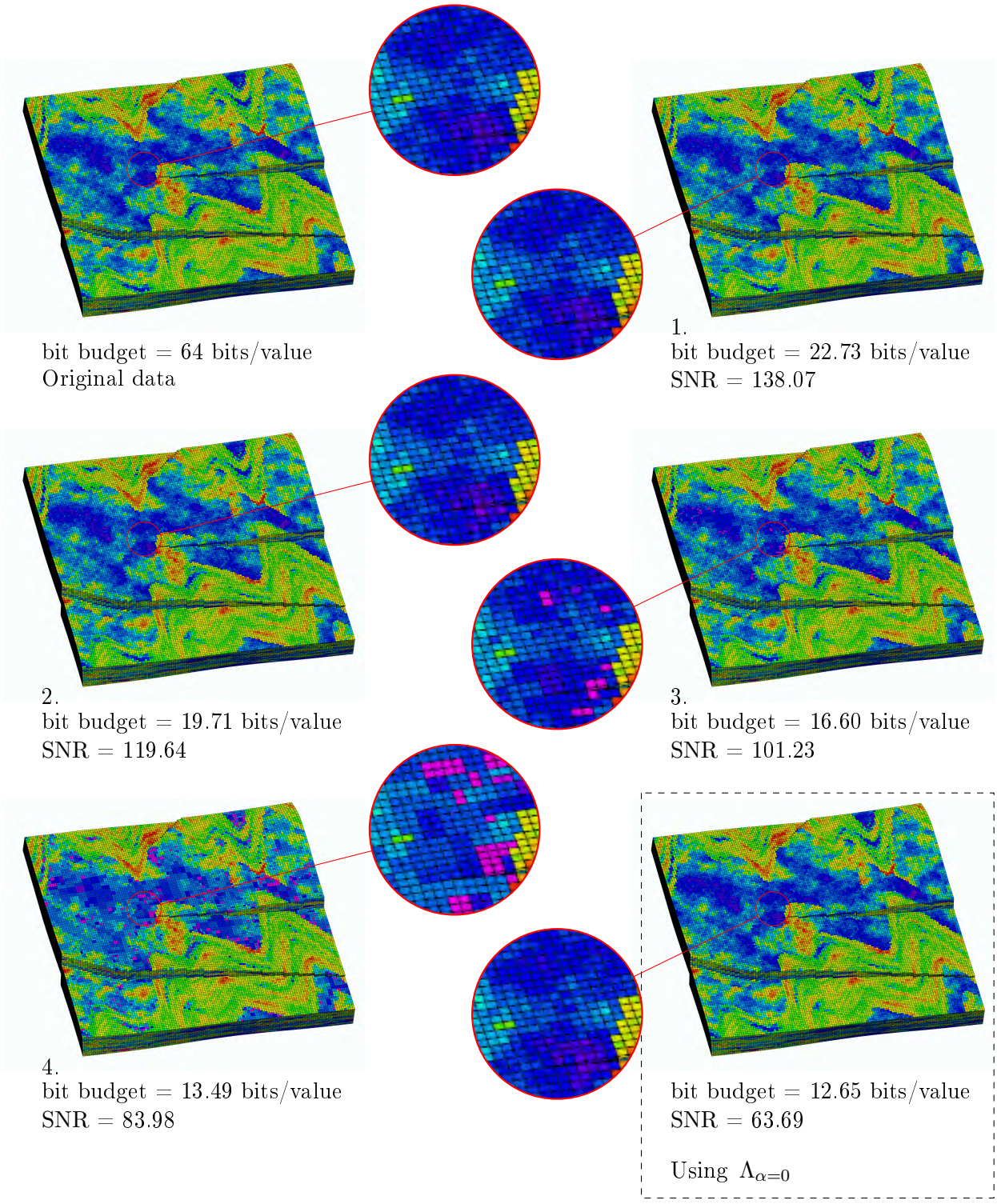


Figure 3.23: Original fluvial permeability property generated at refinable precisions by using lower bit budgets of data (preprocessed by Λ) processed by Haar+ZT combination. Quality is evaluated here by SNR metric. Precisions noted from 1 to 4 correspond to the red circles on Figure 3.22.

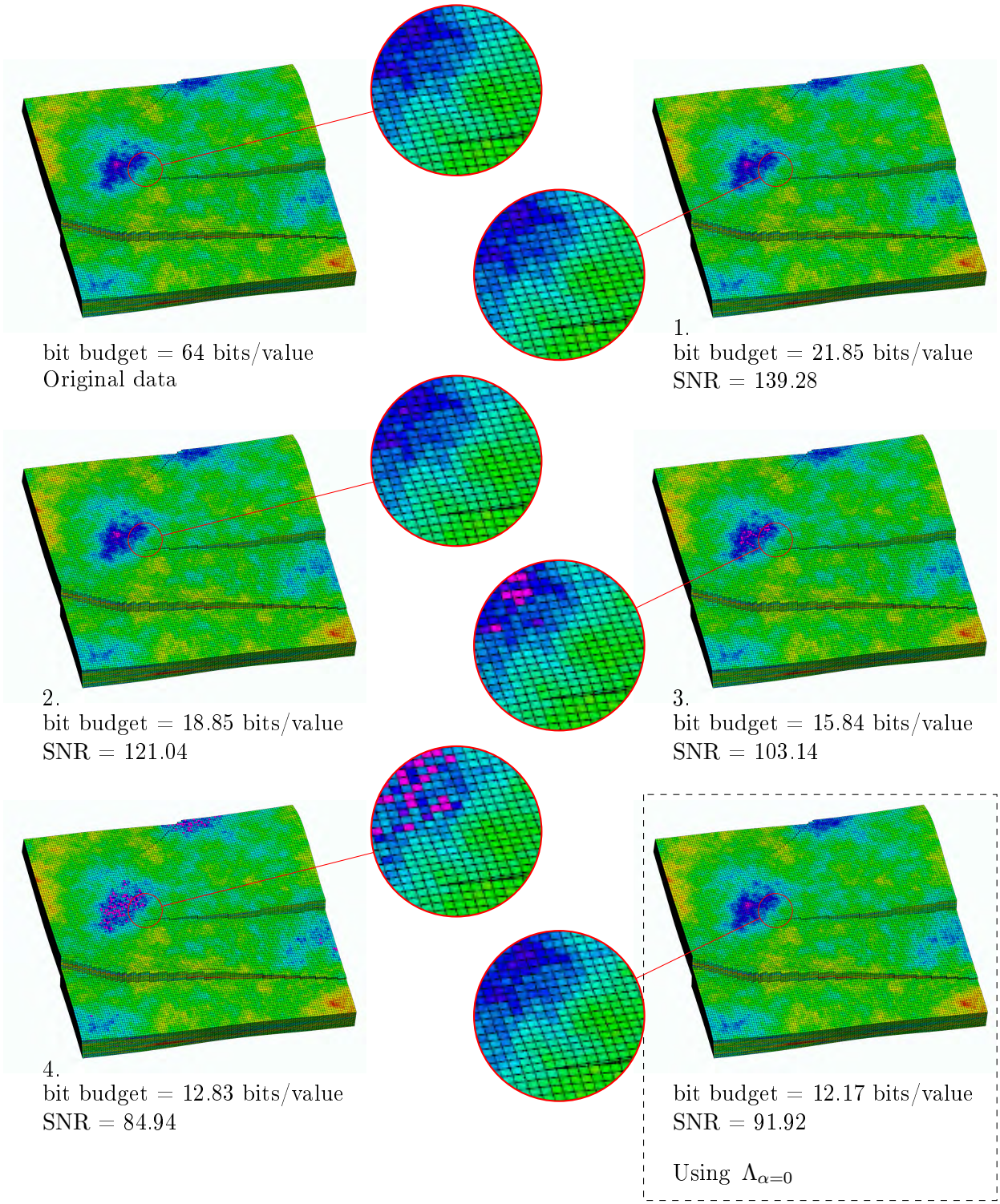


Figure 3.24: Original nearshore$_0$ permeability property generated at refinable precisions by using lower bit budgets of data (preprocessed by Λ) processed by CDF 9/7+ZT combination. Quality is evaluated here by SNR metric.

but the efficiency depends on the encoder and on the meshes features. As the mesh is heterogeneous, we also detailed the compression performance according to the mesh components. We noted a general improvement compression by using HexaShrink whatever the component, except for the continuous properties. For this data, HexaShrink combined to an encoder slightly increases the size of the encoded data, compared to using an encoder alone. The reason is that the continuous properties are floating-point data, with a high precision and distributed on a wide range, which makes their compression more challenging.

The second contribution was to experiment an evolved encoder as the ZT coder to better exploit the multiscale representation provided by HexaShrink. One conclusion is that the HexaShrink decomposition developed for continuous properties is not the most suitable for compression, although fully reversible and exact. Therefore we replaced it by classical $3D$ wavelets, and associated them with success to the ZT coder. We showed that compression at refinable precision is possible, by gradually decompressing the data, which permits to reach bit budgets significantly lower than with generic lossless encoders. Finally, our experimental results showed that, even at limited precision, the visual degradation of the mesh renderings can be negligible.

In next chapter, we continue to demonstrate the benefits of our data representation, showing in particular that a refinable precision has a limited impact on a simulation workflow.

CHAPTER 4

Impact on simulation performances

"It is not about the loss of information, it is about doing more science!"

John Dennis, 2013

Lossy compression has progressively integrated simulation. For few years now, it is used for visualization of large simulation outputs, and only recently first tests combine it with calculation to relieve the workflow. However, whatever the step "compressed" in the workflow, it is primordial to evaluate its impact on simulation outcomes. While there are many metrics for visual assessment none metric objectively assess the impact on simulation since expectations differ according to the field. To optimize the compression in simulation, we thus need to understand the requirements of professionals in geoscience. The discipline uses the simulation results to validate any change in simulation parameters. For this study, we refine the precision or resolution of our mesh properties thanks to HexaShrink. The quality of the refined data is evaluated upstream with objective metrics, and verified downstream after simulation with its results and a subjective validation made by engineers.

4.1 Compression in simulation

In this chapter, we study the impact of refinable precision on simulation, both for spatial and numerical precision. We especially focus on the impact of numerical precision induced by lossy compression in our literature review than on data reduction with resolution change. Although the latter is a standard but crude method in science to accelerate the computation process (by subsampling, such as upscaling/upgridding), it has recently been shown that data reduction has a impact on analytical results worse than compression, at equivalent binary quantity. This observation motivates our interest for lossy compression for simulation. The following review, mainly related to studies in the fields of fluid mechanics and climatology, confirms our intention to quantify spatial and numerical precision of simulation data.

Until few years ago, scientists were suspicious about the use of lossy compression during a computational process, fearing the loss of significant data. Usually accepted in the context of visualization, it was contested for simulation and analysis. But this is now changing. Slowly, in conjunction with the exponential increase in digital consumption, researchers are becoming aware of the growing difficulties with bottlenecks and the increasing costs of computing resources reported in Section 1.2. Baker et al. (2014) explain that daily-generated simulation results are significantly under-exploited considering their cost of production, due in part to limited storage capacity.

Lossless compression first appears as a suitable solution, by reducing the data quantity while preserving the exact accuracy of processed data. Unfortunately, this conservative method provides poor performance on scientific data. The compression ratio indeed barely exceeds two (Son et al., 2014) because of the high precision and the dynamic range of numerical data. Besides, it is known in multimedia that lossy compression allows improving significantly compression ratios, with mostly undisturbing perceptual changes. Accordingly, our objectives in this chapter are to address the effects of compression on simulation workflows and to understand to what extent its impact may be acceptable in simulation. Finally, we question the need to maintain the high precision of simulation data.

To illustrate the heavy process of simulation, we describe the Computational Fluid Dynamics (CFD) approach, used in our case study (*cf.* Subsection 4.2.1). Similar to geosciences, other fields using simulation are strongly impacted by processing huge data quantities, especially in climatology. Here, compression is increasingly looked at, as evidenced by the sharing of large datasets in open source to study the impact of lossy compression. However, the majority of studies still use compression more to improve visualization and storage than to integrate it into their workflow. Our review focuses on integration work and classifies the studies according to their fields, **databases** and **codes**. Besides, part of the review refers to quality measures commonly used to study the impact of compression. Subsection 4.1.2 is the core of our review: compression and evaluation methods are described in an applicative context. We note a growing use of compression and the establishment of routines in the choice of evaluation tools and methods. These routines are mostly borrowed from the research teams that develop the most popular compression tools for scientific data, namely SZ and ZFP. Referencing their studies can help tracking additional work and identifying innovative thinking.

4.1.1 Application for fluid dynamic

Simulation is a numerical tool used by researchers and engineers to study and generate multiple cases, in order to better understand a complex system (gas injection, combustion, turbulent fluid behavior *etc.*). It may provide a detailed vision of an inaccessible phenomenon, at different levels of resolution.

Its use extends to various fields, ranging from engineering design (vehicle, aerospace, *etc.*) to the study of the formation of the universe, using molecular dynamics (MD). At a smaller scale, MD can also simulate the behavior of materials and chemical reactions. But as mentioned in the introduction, our scope of interest is more focused on geosciences and in particular on applications using CFD for simulation. Biology domain also use simulation resources to learn about gene expression by simulating probabilistic and statistical events.

Fluid dynamics example

The study of fluid motion by CFD in a limited volume involves solving systems of partial differential equations (PDEs) using numerical methods as finite difference, finite volumes, finite elements, or spectral methods. The scientific domains using CFD differ by the nature of fluid and the context, such as climatology considering an air volume (atmosphere) for weather forecasting or climate change studies. Projects in this domain are carried out by leading international laboratories like NCAR, because the atmosphere is a complex area in interaction with adjacent layers, defined in the introduction of Chapter 2.

In a straightforward case of fluid dynamics, as laminar flow into cylinder pipe, the issue could be analytically solved. It means the exact solution can be computed for any given point in the control volume, with pressure and velocity values as results. But considering complex problems, an analytical solution does not necessarily exist. In this case, we try to approximate the flows by a physical law formalized by parameterizable mathematical equations (Navier-Stokes equation for the study of turbulent flows, Darcy equation for the study of flows in porous media). The most complex situations may require additional components. For example, the modeling of the transport of chemical elements in the atmosphere (such as pollution) additionally uses an advection-diffusion equation, which exponentially increases the numerical and temporal cost of resolution. For reasons of economy, the modeling of a system of equations is the result of a compromise between representation accuracy and computational resources.

After modeling the system of equations, the solution is calculated on the control volume. To obtain results at different points, the volume is discretized into small blocks (cells introduced in Subsection 2.1.2). The physical parameters of fluid (velocity, pressure, *etc.*) are calculated for each cell. Their values are conditioned by the values of nearby cells interacting with each other. The solution would certainly be more accurate by using an unstructured mesh but the structured case remains an attractive option because the mesh generation, as well as simulation code, are simpler. Finally, CFD obtains a discretized approximation of the real flow relevant to scientific exploration/analysis.

Depending on the size of the mesh and the precision required, the size of the output of large simulations could reach several terabytes or even petabytes, knowing that the time dimension multiplies the volume of data by the number of time steps. Furthermore, the number of outputs (velocity,

pressure, etc.) tends to increase with the complexity of the system of equations discussed earlier in this section.

Selected domains

As mentioned above, different scientific fields use CFD simulation. The fluid mechanic is the parent discipline. It includes several applicative domains (such as climatology, branches of geology, etc.) studying fluid flows in various volumes of interest (from the atmosphere to reservoir formations). To solve the data processing problem and optimize the use of computing resources, several fields now embrace compression.

Tables 4.1 and 4.2 summarize information pertaining to datasets founding in the literature: domains, format, size and code function. The names of these data are written ***in bold****.* In the field of fluid mechanics, a first comment is to note that the majority of them have sought to integrate compression into computational codes (such as **LULESH**, **Nek5000** *etc.*) instead of applying compression to databases (**Johns Hopkins Turbulence**). Fluid mechanics being the mother discipline, we can foresee that the integration of compression into simulation is an ultimate goal, from which the findings will later be applied to other areas. As a second comment, looking at the example of **CMIP** in Table 4.1, we observe an increasing evolution of volumes generated within the same project over time. From 2000 to today, the few terabytes of scientific data have become several petabytes.

Climatology - oceanography Currently, the simulation domain the most affected by issues in data management is climatology. Unsurprisingly it is the most interested by compression, given the growing number of publications. Simulations made in the domain are meant to understand past, present and to forecast future climates. Studies mentioned in the review are affiliated to well-established computing centers: *National Center for Atmospheric Research* (NCAR) in Denver, *German Computing Center* (*Deutsches Klimarechenzentrum*, DKRZ) in Hamburg and the weather machine at the *Los Alamos National Laboratory*, New Mexico.

Yet, the majority of the examples of simulation data provided in open-access are from NCAR. Among their projects, we can cite **ASR** [Artic System Reanalysis], and **POP** [Parallel Ocean Climate] (initiated by Los Alamos Laboratory and revised by NCAR) meaning for compression. The project the most often encountered in this literature review is unquestionably the **CESM-LE** [Community Earth System Model-Large Ensemble] simulation code. The generated data are open-source, and the results of its analysis are discussed during IPCC *Intergovernmental Panel on Climate Change* (GIEC [Groupe d'experts intergouvernemental sur l'évolution du climat] in french), which sets every four or five years. This gathering of scientists aims at alerting public opinion on climate change. Nevertheless, the cost of simulation and the energy required to transfer/store large volumes of data are underestimated and contrary to claims. Data are indeed likely to be exchanged and copied many times since they are freely accessible to maintain transparency on their analysis.

CESM-LE produced in 2005 2.5 petabytes of raw data, post-proceeded to obtain 170 terabytes, which compose **CMIP Phase5** (Meehl et al., 2005) in netCDF format. While **CMIP Phase6** (Eyring et al., 2016) is estimated to be between 20 and 40 petabytes. To give an idea and visualize what its storage implies, consider a standard external hard drive whose capacities are between 2 and

4 terabytes for a thickness of about 2 cm. To store the largest version of **CMIP Phase6** it would be necessary to pile ten thousand disks of 4 terabytes, with a stack height of 200 meters.

There are several versions of **CMIP** (or side projects: **CESM-ATM, ASR, CAM-SE**). The set represents the most tested data to assess compression. Allison Baker, notably, is one of the first researchers who investigated the subject of lossy compression in climatology. She tested varied tools and encouraged the development of methods inspired by multimedia. Lossy compression tools (such as fpzip, ISABELA, GRIB2, wavelet-zerotree) were used in Baker et al. (2014) to question the relevance of over-precision when transmitting, storing and analyzing data. In this article, she notably underlines the sensitivity of complex scientific codes. With different supercomputers, **CESM** code "will not give the same bit-for-bit results". Therefore, it seems relevant to ask what is interesting in preserving the full precision of the data. Because it appears that limited changes were tolerable, results are subjectively evaluated by experts. It motivates Baker et al. (2017) to pursue this line of research looking to define acceptability by using the subjective opinion of climatologists.

SWOT [Surface Water and Ocean Topography mission] deals with experimental data registered by satellites (NASA, CNES) to provide information on large rivers, lakes, and reservoirs. Although the data are not generated by simulation, we mention his mission because it is a current project of large dimension. Moreover it served to assess compression performance of generic filters available on netCDF format (Delaunay et al., 2019).

Other meteorological objects, such as hurricanes, are tested to assess compression impact (**ISABEL**). Hence datasets in climatology and meteorology generally consist in $2D$, $3D$ or $4D$ grids modelling the evolution of the global or local atmosphere. Temperature, pressure, humidity parameters and velocity vectors compose the information stored in grid cells.

With much fewer public attention than climatology, other domains generate equivalent or even greater data volumes. This review aims at broadening the scope by considering additional domains and projects in cosmology (**HACC**/**NYX**), quantum computers for chemistry.

Geology The domain is rich and comprises several sub-disciplines using simulation. In reservoir engineering, our case study, flow simulation studies the motion of phases (gas, oil and water) in porous media as detailed in Subsection 4.2.1. Furthermore, fluids can also be the geological materials, as it is the case for convective process in Earth mantle/core or magneto-convection in the outer core (Schmalzl, 2003). If the task is to study the rupture of geological material, other simulation approaches than CFD could be used and focus on small objects, such as drill cores (Bouard, 2017) or on larger objects with earthquakes. While, for the moment, little work is still devoted to the use of lossy compression for simulation, such a practice is more frequent for the transmission and storage of seismic data.

An important part of the publications listed here results from collaborations with simulation projects. Some datasets may be confidential and unreferenced. For the sake of repeatability, we listed only works whose data origins are known and referenced. The ones marked by an asterisk (*), are referenced by Scientific Data Reduction Benchmarks[1] recently constituted to assess compression methods. The initiative in the context of Exascale Computing Project came mainly from Argonne and Livermore laboratories teams, leading actors on the subject.

[1] https://sdrbench.github.io/

Domain	Name	Format	Size	Used by
Fluid Mechanic	Johns Hopkins Turbulence Database 2008	HDF5	400-440 TB	Treib et al. (2015) Ballester-Ripoll et al. (2019)
Climatology	CMIP Phase 3 to 6 Coupled Model Intercomparaison Project	netCDF	Phase2 (2001) = 0.5 TB Phase3 (2007) = 36 TB Phase5 (2013) = 2.5 PB Phase6 (2016) = 20 to 40 PB	Underwood et al. (2020) Zhang et al. (2019) Baker et al. (2014, 2016, 2019) Chen et al. (2014) Yuan et al. (2016) Cappello et al. (2019)
	CESM-ATM*		1.47 GB + 17 GB / 1.5 TB	Tao et al. (2019)
	Hurricane		62.4 GB	Tao et al. (2019)
	Hurricane (ISABEL*)		1.25 GB	Underwood et al. (2020) Cappello et al. (2019)
	NCAR data archieves	-	-	Welton et al. (2011)
	CAM-SE2011			Zender (2016)
	Layer Packing Tests 2016	netCDF4		
	ASR	netCDF4	v1 (2000-2012)= 10.19 TB, v2 (2000-2016) = 53.09 TB	Yuan et al. (2016)
Oceanography	POP 2002			Woodring et al. (2011)
	SWOT	netCDF	TB per day	Delaunay et al. (2019)

Table 4.1: Datasets

Domain	Name	Function	Used by
Fluid Mechanic CFD	LULESH 2012	Shock hydrodynamic mini-app	Laney et al. (2014) Li et al. (2017a)
	Nek5000 2008	Spectral element CFD solver	Otero et al. (2019) Calhoun et al. (2018) Zhang et al. (2019)
	PlasComCM	Multi-physics Tubulent Flow	Calhoun et al. (2018)
	Miranda *	Hydrodynamics code Navier-Stockes	Laney et al. (2014) Ballester-Ripoll et al. (2019)
	Fuel jet 2011		Ballester-Ripoll et al. (2019)
	Cubism-MPCF 2013	Cloud cavitation simulations (bubble)	Hadjidoukas and Wermelinger (2019)
Climatology	NICAM	Non-hydrostatistic icosahedral atmospheric model	Sasaki et al. (2015)
Cosmology	HACC *	Hardware/Hybrid Accelerated Cosmology Code	Cappello et al. (2019) Underwood et al. (2020)
	NYX *	N-body/hydro code	Tao et al. (2019) Underwood et al. (2020)

Table 4.2: Simulation codes, mini-app

4.1.2 Impact of compression on application

The usage of lossy compression through simulation workflows became popular in the last five years. This practice knows an increasing success with the introduction of new lossy compression tools for scientific data, firstly with ZFP (Lindstrom, 2014), followed by SZ (Di and Cappello, 2016; Liang et al., 2018b; Tao et al., 2017), already introduced in Subsection 3.1.3. Behind these projects are research teams on compression from Lawrence Livermore National Laboratory (LLNL) and Argonne National Laboratory (ANL), led respectively by Peter Lindstrom and Franck Cappello. They produced a large number of publications to test and compare performances of compression tools on scientific data from diverse domains.

As an evidence of the growing interest for the topic, since 2016, more than a dozen of graduate students joined ANL team to work on SZ projects, to apply and adapt the method to various steps of simulation for different domains. They contributed to publishing more than twenty papers at various IEEE Conferences on Big Data - for High Performance Computing, Networking, Storage and Analysis - Parallel Distributed processing Symposium *etc.* mainly dedicated to HPC. The reference article of Cappello et al. (2019) compiles seven lossy case studies and points the main simulation bottlenecks (listed in Subsection 1.2.2, and denoted in *italics* in the rest of the section). By identifying the specific needs in rate, ratio, and accuracy for each application, they promote the notion of a specialized compressor to meet particular requirements.

To *accelerate the execution* of quantum simulation in chemistry, Gok et al. (2018) adapted SZ into SZ-PaSTRI. They improved SZ prediction by pattern identification. In the same vein, Liang et al. (2018b) considerably ameliorated compression performance on cosmology simulation (**HACC**) by reducing *storage footprint*. They combined SZ with a logarithm mapping transform and changed the tool name into SZ_vlct. The method was finally incorporated into SZ 2.0 (selecting point wise relative option). In heavy simulation, *checkpointing* consists in creating periodically temporary files to backup intermediate steps in case of simulation failure. Lossy compression is there used to reduce file size, in order to relieve the overall process without degrading simulation results. This simulation step has been the subject of many works from leading laboratories, testing the impact of lossy compression on temporary files. In the example of Calhoun et al. (2018), at ANL, SZ is used in **PlasComCM** code to reduce the overall checkpointing time.

Most recently, Underwood et al. (2020) developed FraZ to optimize existing tools (SZ, ZFP, MGRAD). Since compression codes are rich in options, performance varies depending on option selection[2]. To reduce these differences, FraZ optimizes the fixed rate option, which is less efficient, compared to the fixed accuracy. Affiliated to ANL, we note that simulation data used for the work are shared at the Scientific Data Reduction Benchmark (such as **Hurricane, HAAC, CESM, NYX**).

From LLNL, Diffenderfer et al. (2019) theoretically established the bound of the error introduced by ZFP integrating operators during its compression. Continued by Fox et al. (2020), the method theoretically addresses forward and backward error for simulation and validates the process by using PDEs that model diffusion and advection phenomena. We note that Lindstrom et al. (2016)

[2]It is, therefore, necessary to have a good knowledge of the tool to obtain the best compression: what is difficult to ask from users of non-computer science backgrounds.

collaborated on an isolated work related to geology. In this field, compression methods are familiar because they have been historically used on seismic data (but barely for simulation, as already said at the end of the Subsection 4.1.1). This time, instead of applying the method to the experimental data, the compression is applied to an operator used for full 3D seismic waveform tomography (f3dt) simulation.
Beyond the studies carried out by LNL and ANL, SZ and ZFP are commonly used as benchmarks in other laboratories to compare their performance with their proper compression tools. Zhang et al. (2019), for instance, compare SZ & ZFP to a method based on block decomposition supplemented by DCT and followed by adaptive quantization. They demonstrate the viability of their solution to restart a simulation from a lossy state, by checking error propagation. The method is applied with success to **CMIP5** and **Nek5000**. As seen with the previous example, the rest of the section deals with tools based on wavelet transforms or using similar decompositions to integrate it into their workflow. SSEM, for instance, is a wavelet-based method developed by Sasaki et al. (2015) during a collaborative work between Tokyo Institute of Technology and LNL to deal with checkpoints for a climate application: **NICAM**. In another study, **Nek5000** is again used by Otero et al. (2019) to simulate a turbulent pipe, or the effect of wind on a wing. They apply a compression based on the Chebyshev transform to the outputs. This preliminary study visually evaluates the impact and invariance of results through an statistical analysis. Otero et al. discusses the fact that 70 % of the data could be truncated, while preserving accuracy for scientific needs. In the article conclusion, compression for checkpointing/restart figures as future works. Indeed, this step represents an accessible bottleneck in the workflow, easily optimized: by reducing the quantity of processed data, the simulation becomes faster. The ultimate purpose would involve integrating the method into the simulation code to optimize, for instance, the memory allocation. But for the moment, the majority of studies are examining the feasibility of lossy compression by visualizing and quantifying its impact through objective metrics.

Objective evaluation

We now focus on the methods used to evaluate the impact of lossy compression in simulation. As with image and video compression, the impact on simulation can be assessed using conventional objective measures. However, these measures may not be relevant since they are not adapted to the scientific application.

To optimize scientific data visualization, the TTHRESH project developed by Ballester-Ripoll et al. (2019) evaluates the impact of compression on volume data from the **Johns Hopkins turbulence database**. The choice of classic objective measures (such as relative error, RMSE, and PSNR) seems relevant if it is limited to a visual assessment. The same argument is questionable when considering simulation data. Therefore it would be wise to “assess the effects of data compression in simulations using physically motivated metrics", as stated by Laney et al. (2014). Their study uses, among others, fpzip compression, and integrates it into three simulation codes. Among them, there are a Lagrangian shock-hydrodynamics code from **LULESH** and an Eulerian higher-order hydrodynamics turbulence modeling from **Miranda**. Metrics motivated by physics are symmetry shock radius for the first case, and Rayleigh-Taylor instability and spectrum of perturbations in the second case. This work is a landmark study, being the first “applying lossy compression to the

physics state of simulations as a strategy for mitigating the data movement bottleneck expected on future systems". He considers that compression ratios under 3:1 do not affect the simulation results. The usage of greater compression ratios may be dependent on applications.

Nevertheless, it is undeniable that objective metrics are easily implemented and fast to compute (because roughly computed from point to point differences as explained in Subsection 1.3.3). It appears resourceful in this case to combine such evaluations with more costly physical analysis methods (introduced by Laney et al. (2014) for instance) and check their correlation. For example, to determine the optimal wavelet kernel for turbulent flow visualization, Li et al. (2015) in a comparative study evaluate classical error metrics such as nRMSE in parallel with the enstrophy. This parameter, in fluid mechanics, computes the kinetic energy by considering the volumes of the highest values. In the same vein, Pulido et al. (2016) compare the impact of different multiresolution transforms (B-splines, Daubechies, Coiflet, curvelet, surfacelet *etc.*) and evaluate alterations on turbulence datasets by considering vorticity, isosurface, curvature *etc.* in parallel to classical PSNR. Such a crossing between different evaluation methods is classical in multimedia. By combining classical objective metrics and subjective evaluations (like MOS defined in Subsection 1.3.3), we could obtain reasonable objective thresholds for which a subjective alteration is acceptable for human perception. For scientific data, a subjective evaluation depends on the competent opinion of specialists who interpret the quality of (de)compressed data.

Subjective evaluation

To subjectively assess the performance of several compression tools, Baker et al. (2019) collaborate with climatologists on the project **CESM-LE**. Four distinct variables are compressed ($2D$ surface temperature, $2D$ clear sky net solar flux, $3D$ grid box averaged cloud liquid number, $2D$ convective precipitation rate) at refinable precisions. Thereafter, they are subjectively evaluated by climatologists, who are asked whether the data are “the same”, or different from the original. To identify the relevant objective metrics, i.e., correlated with expert opinion, ten objective metrics are tested, including MSE, PSNR, SSIM (as defined in Subsection 1.3.3). For this study, SSIM obtains the closest results to the subjective opinion of specialists. The SSIM threshold required to preserve the data precision in this study is higher than the SSIM required for medical data to ensure diagnosis (Baker et al., 2017).

In conclusion to this state of the art, we demonstrated that compression in simulation is a relatively recent concern. Although the publications on this topic are still in limited number, they are carried by renowned laboratories, and the number of related papers rises. Some works use classical methods (tools, quality assessment) borrowed from multimedia data compression, while others study the domain requirements to propose tailored solutions. More recently, exascale projects in simulation emerged, giving rise to the need to process mass-produced data. The quantity of data to be produced in the future encouraged the development of compression tools for scientific data and the creation of specific benchmarks, highlighting the advent of a new discipline marked by the necessity of rationalizing the simulation data.

Considering this review, our work will be between the works of Laney et al. (2014) and of Baker et al. (2019). Similarly with the first one, we focus on the application of simulation and consider metrics specific to the field of reservoir engineering. We also specify subjective criteria, as in the

second publication, that we crossed with objective methods to validate our compression workflow on an extensive simulation benchmark.

4.2 Evaluation of multiresolution progressive compression method

To evaluate the impact of a lossy compression stage in a simulation workflow, we first design a case study (introduced in Subsection 4.2.1). It includes an input mesh, called LUNDI, which is none other than the mesh#6 in the benchmark introduced in Section 3.3. This mesh has been already processed with HEXASHRINK and generic encoders for lossless compression. This enables us to make the link with a complete simulation workflow.

Subsections 4.2.2 and 4.2.3 present simulation results using our scalable progressive representation. We perform tests according to refinable resolutions and refinable precisions. Results are objectively and subjectively assessed to validate the coherency of the different metrics. Finally, the simulation performance obtained from our compression workflow at refinable precision is compared with those obtained from the two most prominent compression tools: SZ and ZFP.

4.2.1 Proposed fluid simulation workflow

Our simulation workflow represents a typical case of reservoir engineering, as detailed in the three next subsections. First, we describe the input mesh. Second, we shortly present the simulation of liquid phase flow through the reservoir. Third, we detail the simulation results, which are analyzed to optimize the exploitation of the reservoir. Since simulation runs are generally repeated a large number of times for sensitivity studies, the aim is to accelerate their calculation by decreasing the number of cells. The method is validated if the outputs of the simulation are not modified or if the change remains acceptable. To evaluate our representation of the mesh at refinable precision and resolution, this argument is decisive, as explained in Subsection 4.2.1.

Versatile RM

Our RM is called LUNDI[3]. If the global structure of the model has been briefly described for the compression benchmark (*cf.* Section 3.3), it is further detailed in this section. We designed a global subsurface morphology integrating three continuous vertical faults. They are not aligned along grid axes, and possess different offsets, to emulate mildly complex environments. This morphology may also challenge compression algorithms. This model can be discretized with different grid dimensions. For our evaluation, we choose a grid with $128 \times 128 \times 32$ cells to allow reasonable simulation times, with respect to our extensive comparative workflow. The average cell size in this LUNDI realization is about $1.7 \times 1.7 \times 0.95$ m^3. This shape ratio for the tiles is common in (sedimentary) geology for modeling mesh cells. It is consistent with progressive horizontal fine deposits of material over the years.

LUNDI topography arises from a realistic geological context. It models a side of an anticline structure (classical in hydrocarbon trap reservoir study), as spotlighted by Figure 4.1. Therefore the most elevated corner of LUNDI corresponds to the top of the anticline (at 3360 m depth). The

[3] LUNDI refers to the Atlantic puffin in the Icelandic language (*macareux moine* in French), a pelagic and relatively pacific bird in the rather competitive world of Petrel and Skua seabirds, or associated software.

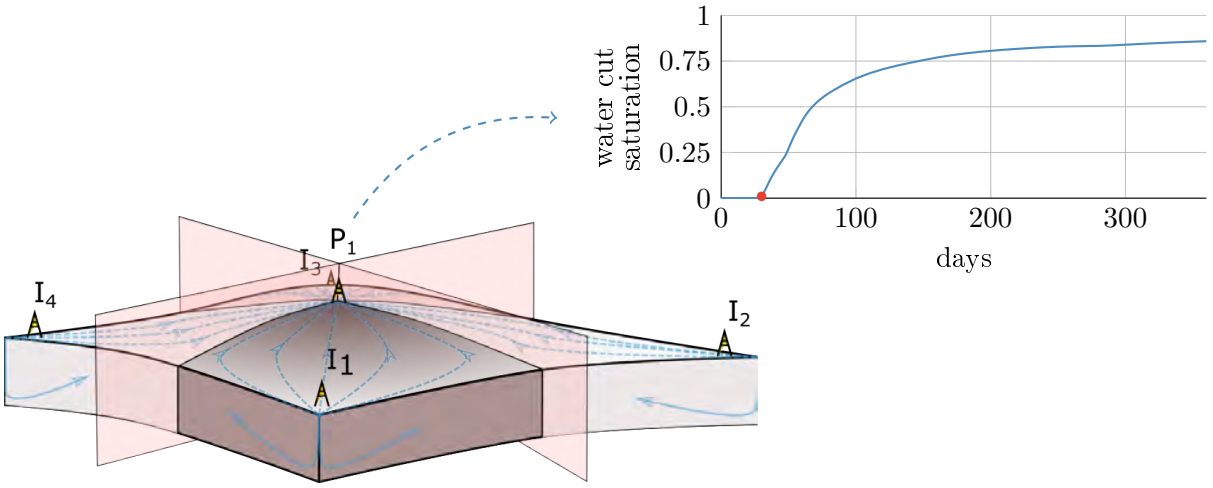


Figure 4.1: (Left) illustration of a quarter five-spot model. Blue dotted lines depict motion of water injected from corners by four wells noted I_i up to producer P_1 on the top of anticline. Our model LUNDI (quarter of anticline) is highlighted in brown and limited by vertical red planes. It only considers two wells (I_1 and P_1) among five. (Top-right) water cut curve measured over a year at P_1. The red dot indicates the water breakthrough.

opposite corner on the diagonal is situated 50 m below. Our benchmark is thus based on a quarter of the anticline modeled by LUNDI. The global shape of LUNDI as well as its structural discontinuities, have been thought as a challenge for multiscale visualization and faithful upgridding for simulation.

Two-phase flow simulation

We use for our simulation the black oil model (Thomas et al., 1976), and limit the study to two liquid phases: oil and water (we ignore gas phase for the sake of simplicity). The system can be described by physical laws (conservation of mass and Darcy law). It can be modeled by a mathematical system composed of non-linear partial differential equations. Finally, unknown system parameters, such as pressure and saturation, are computed for each cell, for each time step.

Two-phase flow simulation is thereafter performed on LUNDI. It is meant to predict oil production in a two-phase reservoir, driven by water injection. At the initial time, the two phases in the reservoir are horizontally stratified, with oil above water. We simulate a quarter five-spot configuration inspired by (Lie, 2016, Chapter 5.4.1). It involves two wells, as shown by Figure 4.1: one producer P_1 and one injector I_1 (among the four ones $\{I_i\}$ localized on the anticline base). Water is injected by I_1 in the lower part of reservoir. The water pressure pushes the oil through the reservoir up to the producer P_1 (distant from 300 m). Injector pressure and producer rate remain constant, respectively set at 300 bars and 300 m^3 per day.

Reservoir production overtime

Oil production is evaluated according to various reservoir properties and injection settings. One of the reference indicators is the estimated water cut (WC), or the ratio of water produced in a well compared to the total liquid volume. It is recorded over a period of time, for a certain time step, and can be illustrated by the typical WC shape in Figure 4.1-right. Furthermore, output analysis can also focus on other parameters known to reservoir engineers, such as well pressure or oil production, to be sensitive to the slightest change in the input data.

WC is a valuable source of information for decision anticipation for field exploitation. The inflection point marked by a red point on the curve Figure 4.1 is the water breakthrough. It denotes the "first" water arrival to the production well. After this date, extracted liquid will progressively contain more and more water. Hence, to avoid eliminating water by expensive post-treatment or surface equipment, this is interesting to delay this date by changing the exploitation method. Simulation is a powerful tool to predict the water breakthrough and to determine the best option, by testing different methods and conditions. Hence simulation can be run a large number of times with varying parameters (reservoir sensitivity analysis). It therefore seems reasonable to limit its calculation cost by using a lower resolution grid.

Well production metric

To check whether standard upscaling is suitable for simulation, reservoir engineers carefully inspect the simulation outcomes and compare them to outcomes obtained with the reference RM. This approach is common to validate the accuracy of a RM at a lower resolution. We extent it to the assessment of compression at refinable precision through a complete comprehensive compression-simulation workflow, illustrated in Figure 4.2. The reference for well production is the WC curve

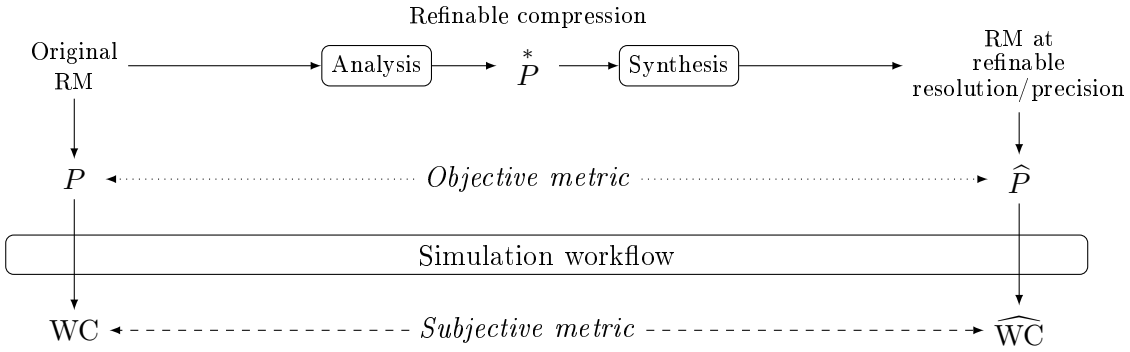


Figure 4.2: Objective and subjective evaluation steps throughout our compression-simulation workflow.

provided by simulation of the original RM. We denote by $\widehat{\text{WC}}$ the water cut curve provided by simulation of compressed RMs. We separately address the impact of a global refinable resolution of mesh (for upscaling/upgridding), and that of refinable precision of properties (here permeability). The differences between WC and $\widehat{\text{WC}}$ curves should be evaluated with respect to the expected

benefits of a refinable precision. Firstly, we evaluate well production subjectively with a qualitative ranking based on reservoir engineering expertise. Secondly, to automate decision of suitable refinable compression, we assess candidate objective metrics, and finally propose adaptations to better predict subjective evaluations on different geological environments.

Subjective evaluation of simulation results Depending on their distance to the reference WC curve, simulated $\widehat{\text{WC}}$ outcomes are categorized into five qualitative classes distinguished by their degree of accuracy. For the sake of rigor and reproducibility, we define dotted lines delineating the five classes. The overall shape is summarized as follows (we refer to appendices for a detailed description, page 122). We expect that the $\widehat{\text{WC}}$ curves, and by implication the class boundaries, stand close to the reference at the beginning of oil production. The day corresponding to the water breakthrough should be evaluated as precisely as possible since it corresponds to essential information during oil production. Later in the production, precise time accuracy is less vital, and regions are allowed to drift further away from the reference WC. Four pairs of boundaries almost symmetric above and below the WC curve therefore delimit five classes defined by colored areas in Figure 4.3: identical, correct *etc.*

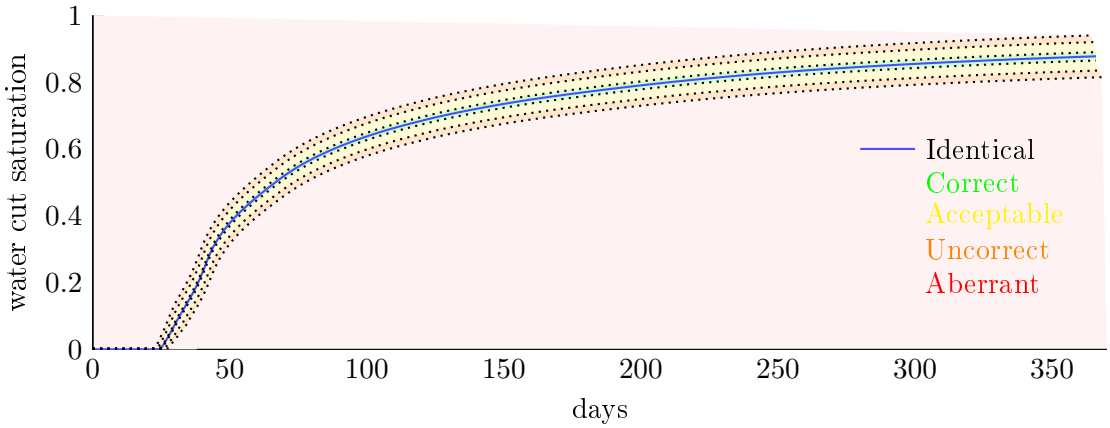


Figure 4.3: Subjective evaluation of simulation results, using region boundaries around the water cut curve (around WC curve from nearshore_0 environment) to define a qualitative ranking: 'Identical' (in blue), 'Correct' (in green), 'Acceptable' (in yellow), 'Uncorrect' (in orange) and 'Aberrant' (in red).

Objective evaluation of refinable representations With HexaShrink, we generate different resolutions of the RM, and then several versions of the RM at full resolution filled by continuous properties at refinable precision. During this work, we mainly focus on permeability for refinable precision.

When considering an upscaling on a RM, several quality indicators, related to petrophysical properties, have been proposed to evaluate the resulting information losses (Preux, 2014). When considering a precision refinement, the conventional objective metrics are classically based on differ-

ences between the cell values of the original property denoted $P(c)$ (c being the indexation of the cells in the mesh) and the cell values filled by a refined property denoted $\widehat{P}(c)$. These differences are generally averaged, and can be considered relative to the original values or squared *etc.*, generating a large number of potential metrics, previously introduced in Figure 1.5 in Subsection 1.3.3. However, they might be ill-suited to petrophysical properties. In the following, we only concentrate on a limited number of objective metrics and propose a novel family of quality indicators combining classical metrics and compandor. The classical metrics we select are: MRE, nRMSE and SNR (respectively on Equations 4.1, 4.2 and 4.3).

$$\text{MRE} = \frac{1}{\mathcal{C}} \times \sum_{c=1}^{\mathcal{C}} \frac{|P(c) - \widehat{P}(c)|}{P(c)}, \tag{4.1}$$

$$\text{nRMSE} = \sqrt{\frac{1}{\mathcal{C}} \times \sum_{c=1}^{\mathcal{C}} \left(P(c) - \widehat{P}(c)\right)^2}, \tag{4.2}$$

$$\text{SNR} = 10\log_{10}\left(\frac{\sum_{c=1}^{\mathcal{C}} P(c)^2}{\sum_{c=1}^{\mathcal{C}} \left(P(c) - \widehat{P}(c)\right)^2}\right). \tag{4.3}$$

The novel family of metrics is obtained by appending the compandor operator lambda to the classical metrics. For instance, the Λ-SNR equation becomes:

$$\Lambda\text{-SNR} = 10\log_{10}\left(\frac{\sum_{c=1}^{\mathcal{C}} \Lambda(P(c))^2}{\sum_{c=1}^{\mathcal{C}} \left(\Lambda(P(c)) - \Lambda(\widehat{P}(c))\right)^2}\right). \tag{4.4}$$

4.2.2 HexaShrink evaluation in upscaling/upgridding

The HexaShrink representation can perform up to 5 levels of decomposition on the lundi reservoir model. The first three resolutions filled by nearshore$_0$ properties (permeability and porosity) are shown on Figure 4.4. Structurally, the faults remain visually coherent across resolutions. The uniformly reduced number of cells across resolutions is detailed in Table 4.3. Properties results of HexaShrink decompositions for our four environments are represented in Figures 5.11 and 5.12 in appendices (page 126).

The two-phase flow simulation is applied on the different lundi resolutions, resulting in distinct $\widehat{\text{WC}}$ curves. They are compared with the reference curve obtained from the original RM. The results in the nearshore$_0$ environment are displayed on Figure 4.5. The results for the three other environments are available in appendices (page 124). First, we observe that the $\widehat{\text{WC}}$ curve simulated on the reconstructed resolution res-0 is identical to the reference WC whatever the environment. This confirms that the HexaShrink decomposition is fully reversible for simulation. Then, we

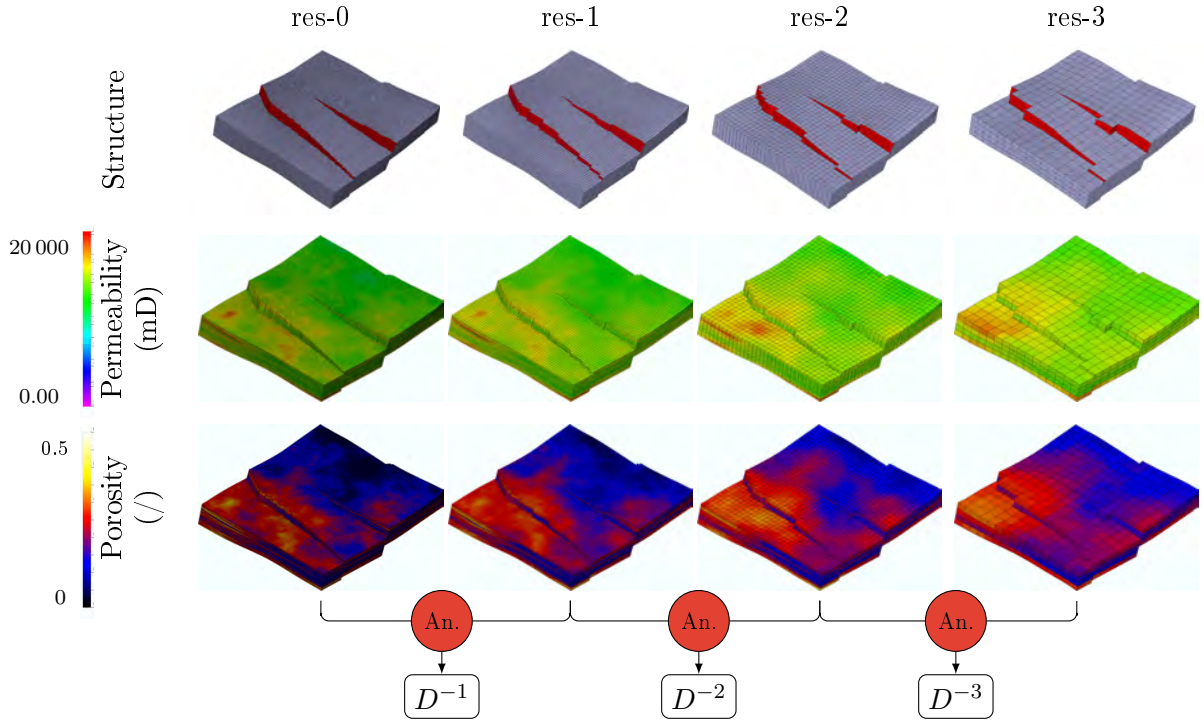


Figure 4.4: (Top-row, left to right) reconstruction of LUNDI and its first three resolutions obtained with HEXASHRINK. (Center and bottom) the same resolutions filled with the permeability and porosity data, themselves decomposed.

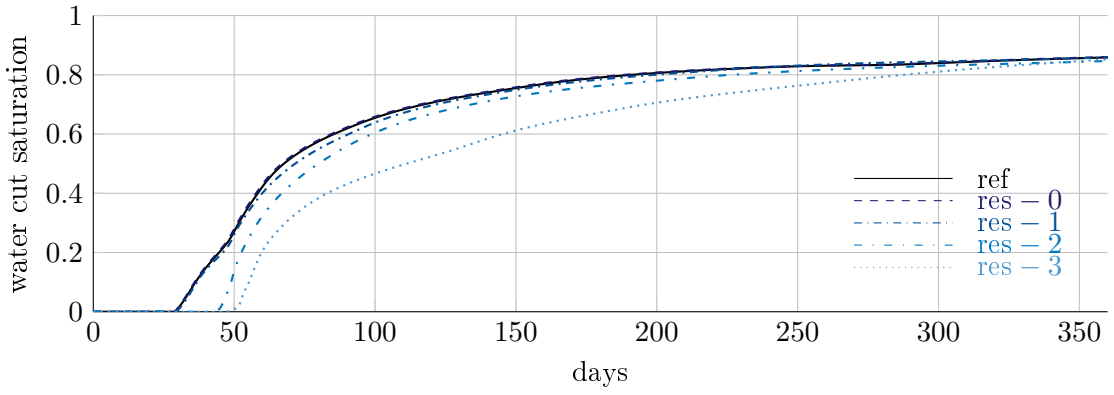


Figure 4.5: Overlay of WC simulated on fine grid LUNDI mesh, (nearshore$_0$ environment) and $\widehat{\text{WC}}$ curves simulated on its lower resolutions (up to res-3) generated by HEXASHRINK.

Resolution	Fine scale model size	Total cell number
0	$128 \times 128 \times 32$	524288
-1	$64 \times 64 \times 16$	65536
-2	$32 \times 32 \times 8$	8192
-3	$16 \times 16 \times 4$	1024
-4	$8 \times 8 \times 2$	128
-5	$4 \times 4 \times 1$	16

Table 4.3: Dimensions of LUNDI and its lower resolutions generated by HEXASHRINK.

observe that the shape of the reference WC curve is modified depending on the LUNDI environment (nearshore_0, nearshore_1, nearshore_a, fluvial). The water breakthrough in fluvial environment happens 30 days after the water breakthrough observed with nearshore_0 environment. Thus, the petrophysical properties significantly affect the simulation outcomes.

Considering the resolution res-1 in the nearshore_0 environment, with a number of cells reduced by eight, its $\widehat{\text{WC}}$ curve stays close to the WC curve reference. Then, according to our subjective metric (explained in Subsection 4.2.1) its $\widehat{\text{WC}}$ curve is "acceptable". Compared to the equivalent $\widehat{\text{WC}}$ curves obtained in three other environments, the outcome obtained with nearshore_0 is the best. This difference may be explained because of the high permeability of nearshore_0 environment (mostly green according to the permeability color scale). In that case, the modification of permeability would have less impact on the simulation. This allows us to focus mainly on the impact of the change in structure due to the use of lower resolutions. We infer that the structure of the mesh res-1 remains suitable to preserve the quality of simulations. Thus, HEXASHRINK is a promising upgridding technique, while further improvements are necessary for property upscaling.

At equal quality, the HEXASHRINK-based simple upscaling is computationnaly efficient. While the original mesh required more than three hours for simulation, only ten minutes are needed with the resolution res-1 on the IFP Energies nouvelles supercomputer Ener440. For comparison purpose, the same simulations run on a scientific laptop (with Intel Core i7-6820HQ CPU @ 2.70 GHz processor and 16 GB RAM) would take much longer: seventy-two hours on the full resolution, and four hours on the resolution res-1.

The results presented in this section highlight the power of upscaling/upgridding in saving simulation time but also the sensitivity of the simulation to the features of the input mesh. A mesh scaling is a complex process that requires care in finding the optimal approach for each component of the RM. From the literature review, we know that the best methods are dynamic approaches presented in Subsection 3.1.1. Unlike uniform merging of cells (eight by eight), the dynamic approach only uses cells outside the flow path to limit the impact on flow dynamic. As efficient upscaling methods use Local Grid Refinement (LGR), we used it to correct the position of the wells.

As specified when describing the simulation in Subsection 4.2.1, the vertical wells cross through the cell columns located at opposite corners of the mesh. Considering the original data, $\{I_i\}$ is located at surface coordinates (1,1). Using lower resolutions, $\{I_i\}$ is still located at the (1,1) but is spatially translated because of cell dimensions increase. LGR method has been applied around the wells to avoid this issue. It retains the full original resolution on cell columns crossed by wells.

However this care does not seem to significantly improve simulation results.

As observed, changes in mesh resolution are not simple. Such manipulations can significantly impact the simulation results. The HexaShrink inherent structure constrains possibilities for upscaling improvement, yet with a simple dyadic decomposition, lower resolutions may provide consistent results and accelerate simulation. In the following, we pursue our quest of essential data required for simulation, we explore the impact of numerical precision and improve the compression ratios.

4.2.3 Continuous properties, which precision?

In the previous chapter (*cf.* Section 3.5), we showed that a ZT coder offers promising compression performance on the continuous properties, in comparison with generic encoders. We now pursue this methodology to investigate the impact of refinable precision on simulation outcomes. In Section 3.5.3, we show that the quality of data compressed with our workflow at refinable precision is evaluated by SNR between 150 dB and 200 dB (see Figure 3.22). It is therefore questionable whether such a quality is adequate and necessary for scientific applications, given that the values dynamic are much higher than those encountered in multimedia. If sufficient, we can further go into compression using lower numerical precisions. Then we will try to define a minimum precision threshold, *i.e.*, the minimum binary quantity (bit budget) for suitable simulation.

Refinable representation with zerotree

Our method is applied to permeability properties. As other scientific data, they are hardly compressible because of their distribution. A specificity of permeability is a large range of possible values (typically from 0.0007 to 20 000 mD as SPE10 model) with a large proportion of very small values. We especially investigate the use of preprocessing permeabilities with compandors (*cf.* Subsection 2.1.3) noted Λ_α. We compare the performance using objective metrics as well as subjective validation through the simulation workflow. *Our principle observations are summarized in italic hereinafter.*

On Figure 4.6 are displayed three versions of nearshore$_1$ permeability at different precisions. The original one and two others only using the eighteen MSB_M of data decomposed by CDF 9/7 wavelet (Cohen et al., 1992) and then encoded with ZT. They differ from the use of a compandor, applied before transformation as a preprocessing. The version on the left uses Λ_0 (logarithmic compandor) while the right one uses Λ_1 (linear compandor). We observe that the left version is better preserved, for instance around the blue patch with low values. On the right picture, the same patch appears noisy with the use of Λ_1 . It concurs that the lower values are better preserved by compression if using Λ_0. Besides, for a similar MSB_M, the overall bit budget is lower with Λ_0, but also the quality is better, visually. This demonstrates that a ZT coder is more efficient on this preprocessed data, better decorrelated by the wavelet transform when the compandor Λ_0 is included in the compression workflow.

- *It appears Λ_0 increases the compression performance of a ZT, and better preserves lower values, visually.*

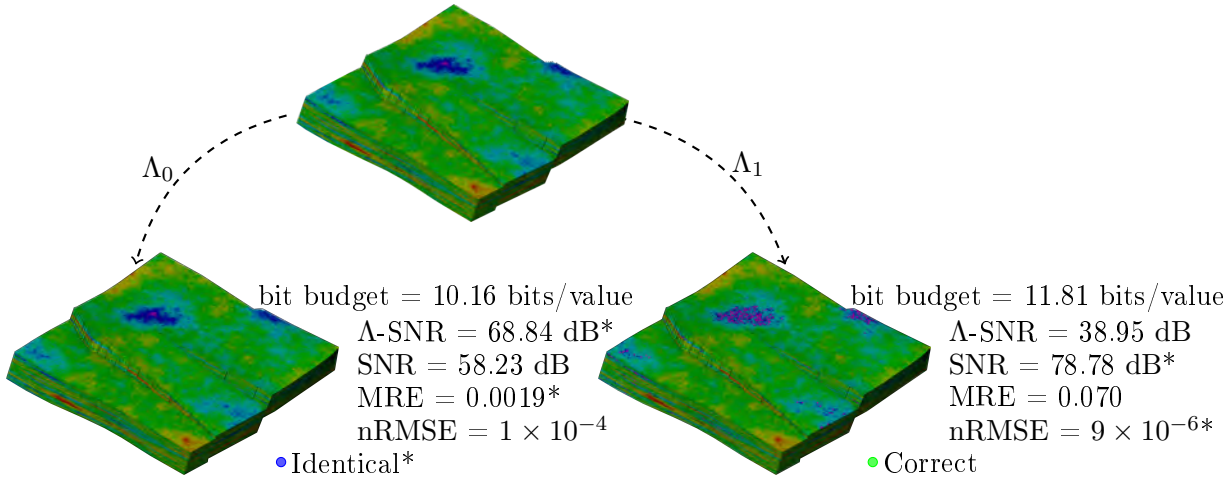


Figure 4.6: From original nearshore$_1$ permeability (top) are decompressed two versions of the data at refinable precision, both using 18 MSB_M processed by compandor: (left) with Λ_0 or (right) with Λ_1 . Data are objectively, subjectively assessed and classical λ compandor objective metrics. (*) indicates the data with the best metric evaluation.

Next, we look at objective metrics whose assessment is correlated to visual expectations and especially to simulation validation, supporting an enlightened use of compression for simulation.

Successively considering the objective metrics MRE, SNR, nRMSE, Λ-SNR, we note several anomalies. As an example, the SNR is equal to 78.78 dB with Λ_1 , and it is superior to the SNR obtained with Λ_0 (58.23 dB). This would mean that the Λ_1 is more appropriate than Λ_0 . Considering our previous conclusions, it finally appears that the standard SNR assessment is contrary to visual appreciation. We note the same anomaly using nRMSE. However, Λ-SNR and MRE indicate the Λ_0 is of better quality, as visually approved. Considering now the simulation results, Λ_0 yields "identical" mention, while the simulation outcome is only considered as "correct" with Λ_1 . Subjective evaluation of simulation results is in this case correlated to visual intuitions. Because simulation is a decisive aspect, we observe that Λ_0 finally generates higher quality data, in terms of visualization but also of simulation. Data quality is foreseeable regarding Λ-SNR and MRE, while SNR and nRMSE provide misleading assessments.

- *Simulation validates visual quality and benefit of the use of the Λ_0 compandor.*
- *Λ-SNR and MRE corroborate visual appraisal and simulation results.*

This example questions the blind use of some metrics (SNR, nRMSE) on scientific data (with their particular distributions), whose assessment could be counter-intuitive.

Figure 4.7 summarizes the MRE, SNR, nRMSE, and Λ-SNR curves we obtained for nearshore$_1$ permeabilities. Dashed curve refers to Λ_1 (linear compandor) while the solid curve denotes the Λ_0 use (logarithm compandor). The bit budget depends on the precision targeted during the (de)decompression, corresponding to a number of bit planes decoded. These graphs also indicates

the subjective evaluation of simulation results with the color code introduced in Subsection 4.2.1. The two compandor versions of permeability displayed in Figure 4.6 are also indicated by surrounded markers on the four graphs, for the four objective metrics.

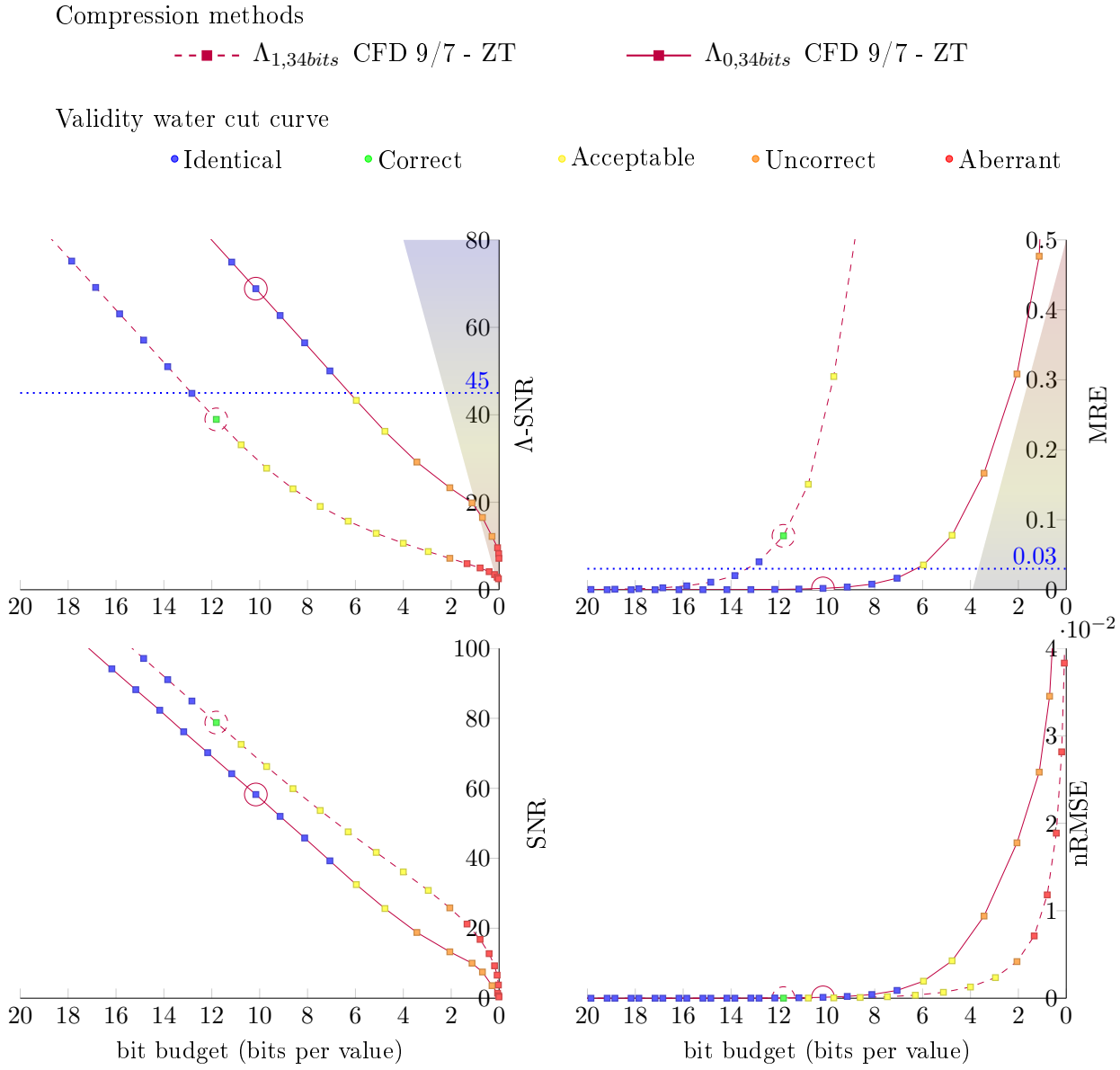


Figure 4.7: Permeability from nearshore$_1$ generated at refinable precision for decreasing bit budget (a mark per precision) by our alternative (changing compandor parameter), objectively evaluated by four quality metrics: Λ-SNR (top-left), MRE (top-right), SNR (bottom-left), nRMSE (bottom-right). Subjective appreciation of simulation results is represented by a color code.

Whatever the objective metric or the compression method, we observe that as expected the global quality decreases with the bit budget (MRE and nRMSE increase, SNR and Λ-SNR decrease). Depending on the relative position of the curves, we can rank the efficiency of the methods. Considering MRE and nRMSE (graphs on the right), the lower curve identifies the best method. For SNR evaluations (graphs on the left), this is the highest curve. As previously noted, the evaluations provided by SNR and Λ-SNR are conflicting despite their mathematical relation. In the same logic, we consider that the Λ-SNR assessment is the most coherent because it is validated by subjective results, while SNR is not consistent with what simulation would suggest. Indeed, regarding subjective evaluation provided by marks color of the SNR curves (bottom-left graph), the simulation results based on Λ_1 are "acceptable" for a bit budget around ten bits. At equivalent bit budget, Λ_0 yield "identical" simulation results. We thus expected that the SNR curve for Λ_0 would be higher, synonymous of better quality. However, it is the reverse. This confirms that the SNR leads to faulty reasoning on these data for reservoir simulation.

Finally, by focusing on consistent objective metrics (top graphs), the higher the Λ-SNR or the lower the MRE, the greater the chances of obtaining suitable simulation results. This seems to provide an objective limit (underlined by the blue dotted lines), below which the data are no longer suitable for simulating with enough accuracy. For a better understanding of the graphics, triangles are attached to the y-axis. They represent a balance between subjective and objective evaluations. Their red tip tends toward poor data quality, while their blue base indicates higher quality.

In the remainder of our report, we only focus on Λ-SNR graphs considered as more readable. Therefore Figure 4.8 keeps the upper left graphic from Figure 4.7 and adds on the right the Λ-SNR graph for fluvial permeability (the results for all the properties are available in appendices). In the same logic as before, a graduated evaluation scale (figured by the color triangle) can be drawn, linking objective evaluation to the subjective one. This allows us to concur that the "identical" threshold varies according to the property and its environment. Considering the fluvial environment, "identical" simulations are obtained for Λ-SNR quality evaluated above 55 dB, while 45 dB is sufficient for the nearshore_1 environment. The difference between the two thresholds can be explained by their level of complexity. The fluvial environment would require a higher objective quality level to preserve channel objects.

In all the studied cases, the use of the Λ_0 compandor enhances the refinable representations of the permeability. Considering the Figure 4.9, we can better perceive this compandor effect. First of all, whatever the value of the alpha parameter used for Λ, the data processed by $\Lambda$are distributed between 0 and $2^{nbits-1}$. If the shape of the distribution remains unchanged using $\alpha = 1$ (linear), histograms are better spread using $\alpha = 0$ (logarithm). Indeed the height of the first bar decreases for fluvial (bimodal: channels and floodplain) and tends to disappear for nearshore_1. The small values, originally concentrated between 0 and 250 mD, are dispersed to several distinct bars. The use of Λ_0 gives them more weight, and preserves them during compression. Concretely, from a physical point of view, it is necessary to preserve the lower permeability values, to save impermeable barriers and to maintain flow. Results for all the other environments are available in appendices (page 130). The "identical" threshold lies between 40 dB and 55 dB. It means that whatever the environment we certify that, above a Λ-SNR equal to 55 dB, simulation results are identical, and those for a considerably reduced bit budget.

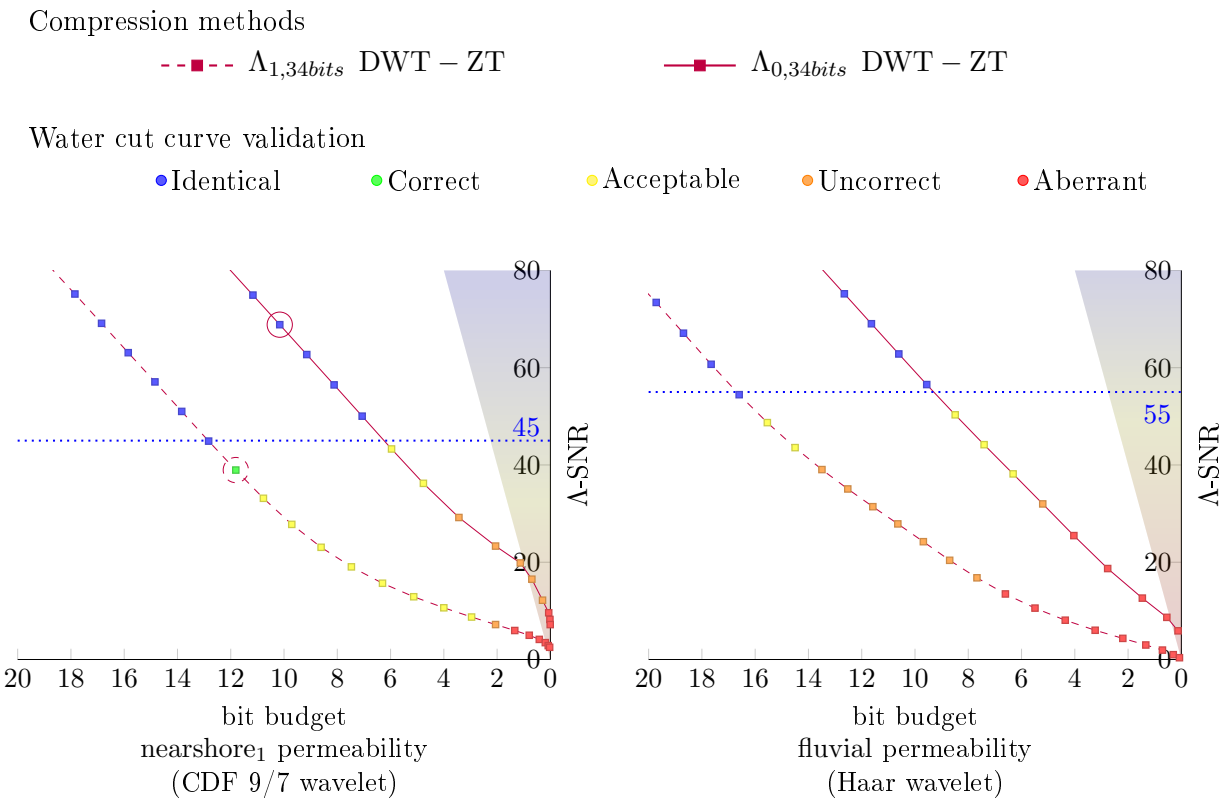


Figure 4.8: Permeability from nearshore$_1$ (left) and fluvial (right) environments generated at refinable precision decreasing the bit budget (a mark per precision) using our method (compandor), objectively evaluated by Λ-SNR. Subjective appreciation of simulation results is represented by a color code.

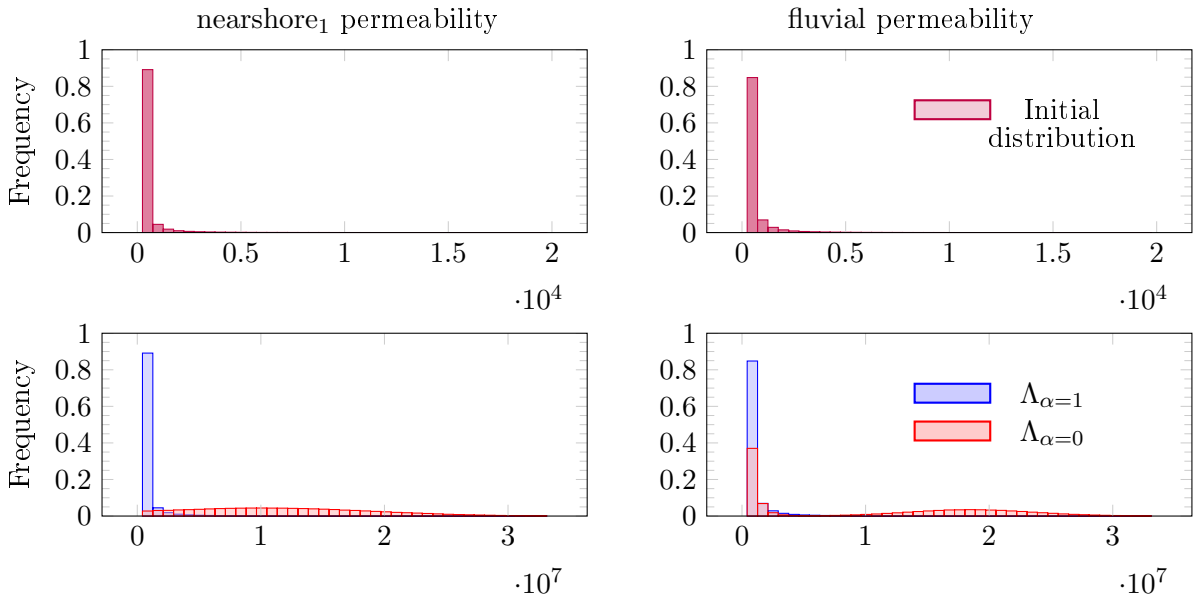


Figure 4.9: Histograms of nearshore$_1$ permeability (top-left) and fluvial permeability (top-right). Initial distribution is between $[0, 20\,000]$, then $\Lambda_{\alpha,25bits}$ using $\alpha = [0, 1]$ distributes the properties between $[0, (2^{25bits} - 1)]$ (bottom).

Comparative study with state-of-the-art

To challenge our method and check the positive effect of Λ_0, we add to our graphs the curves of the probably two most popular compression tools for scientific data : SZ and ZFP (introduced in Section 4.2). We use their recent versions, and compile them with their most efficient options according to (Underwood et al., 2020):

- SZ, 2.1.3 version, with *PSNR* compilation option;
- ZFP, 0.5.5 version, with *accuracy* compilation option.

Results are displayed on Figure 4.10 and complement those presented in Figure 4.8. Dashed lines represent compression results for the regular use of the tools, while solid lines assess the effect of Λ_0 on their compression performance. In addition, the *point wise relative error* option of SZ was also been tested. It integrates a logarithm transform mapping before prediction, and is akin to our Λ_0 approach.

Taking a step back, the results of the other tools confirm the previous observations, notably the "identical" threshold (dotted blue line) at 45 dB for nearshore$_1$ environment, and at 55 dB for fluvial environment, below which the data precision is no longer sufficient for simulation. We observe that Λ_0 , as the logarithm mapping, also improves the compression performance of both compression tools. Focusing on the fluvial permeability (right graph), the data processed by Λ_0 (solid lines) obtain identical simulation results for Λ-SNR above 60 dB. At similar objective/subjective quality, the regular methods (dotted lines) require about 5 bits more per value than the same method using

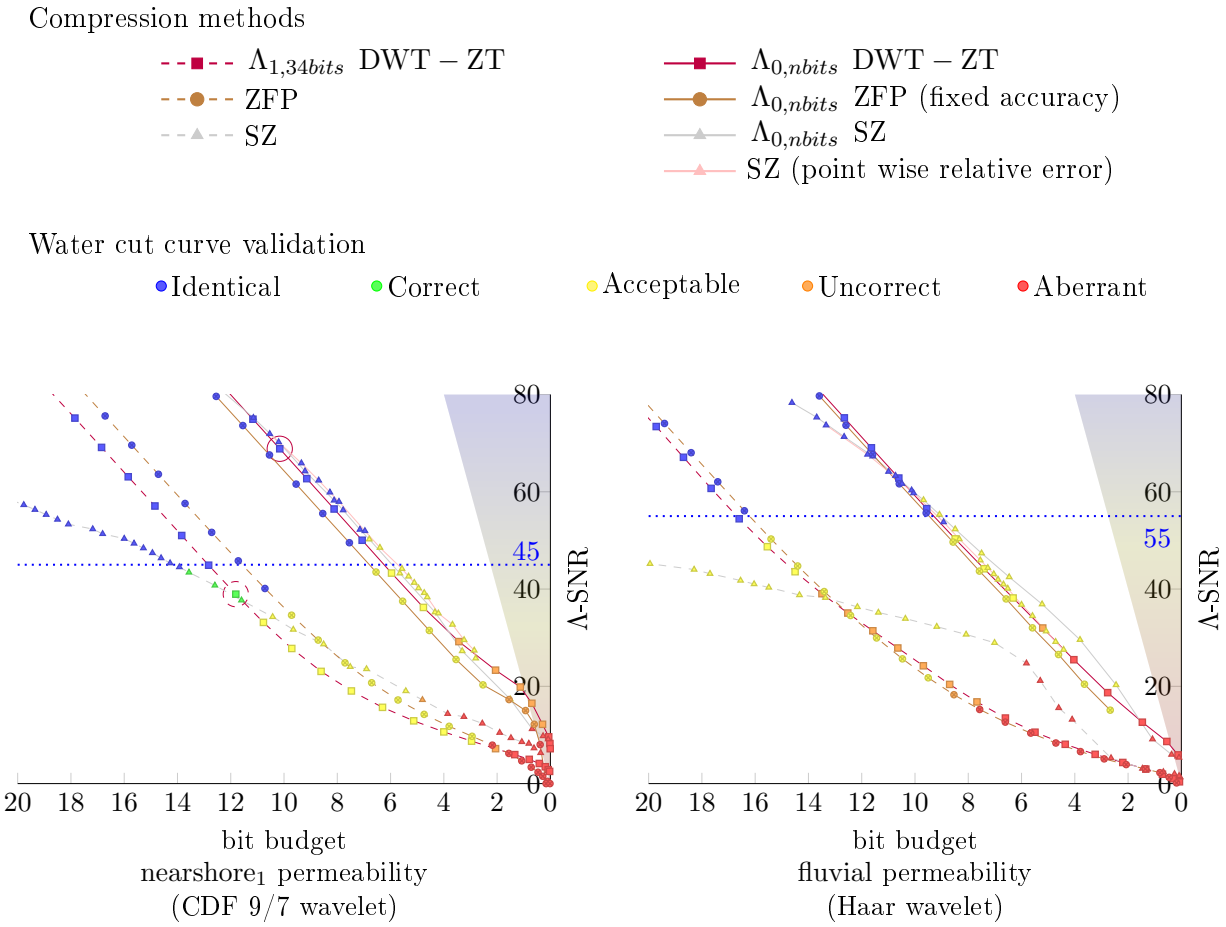


Figure 4.10: Permeability from nearshore$_1$ (left) and fluvial (right) environments generated at refinable precision decreasing the bit budget (a mark per precision) using our method, SZ and ZFP (with or without compandor stage), objectively evaluated by Λ-SNR. The subjective appreciation of the simulation results are still represented by the color code.

a Λ_0, demonstrating the interest of a compandor stage.
We observe that Λ_0, as the logarithm mapping, also improves the compression performance of both compression tools. Focusing on the fluvial permeability (right graph), the data processed by Λ_0 (solid lines) obtain identical simulation results for Λ-SNR above 60 dB. At similar objective/subjective quality, the regular methods (dotted lines) require about 5 bits more per value than the same method using a Λ_0, demonstrating the interest of a logarithm compandor stage.

More specifically, we observe that our method gives results comparable to the other tools (SZ and ZFP). With Λ_1 our alternative is just below ZFP, the best method (without Λ_0 use). Lindstrom's tool is indeed known to be the most efficient on volume data. Like our alternative, it combines a transform with a ZT coder. Thus, it seems consistent to obtain comparable trends.

ZFP gets its superiority from the sub-blocking process explained in Subsection 3.1.3. The equalization of the exponents (floating-point writing) within each subblock absorbs high variations of data scales. This is also one of the benefits of the logarithm function. Therefore combining Λ_0 with ZFP increases the compression performance but not as much as it does for other methods that take the lead. Considering SZ, with Λ_0 we obtain results equivalent to those obtained by SZ with the *point wise relative error* option.

Although not evaluated in term of execution speed for HPC implementation, our approach (regarding its compression performance) provides suitable results comparable to "state of the art" compression tools for simulation. Therefore, incorporating precision layer to HexaShrink constitutes a promising research axis for scalable representations in simulation workflows, including all the components of a RM. The recent paper of Hoang et al. (2021) actually offers an unified encoding of resolution and precision for scalar fields.

To conclude, continuous properties can be generated at refinable numerical precision without visual noticeable degradation, while the simulation outcomes remain "identical" to the reference. To guide the user through the compression process, we identify objective metrics correlated to professional expectations. Our tests show performant compression provides identical simulation results using less than the half of the binary quantity originally used for permeability property (nearshore_1 environment): among 12 bits per value instead of 34 bits. Moreover, the use of compandor functions seems to provide consistent predictions in addressing the properties distribution (here permeability), by better taking into account the physical laws that influence the continuous properties. This example shows that the knowledge of the data (in particular their specific distribution) can considerably improve their processing. Such observations were drawn whatever the geological environment chosen for the simulation of our model lundi.

Lastly, a comparative study with SZ and ZFP tends to corroborate the above observations. There seems to be a minimum quantity of binary information to recover the essence of the RM. This value depends on the data, since our subjective/objective thresholds vary according to the property environment. But even using different tools, there seems to be an agreement to an objective threshold (blue dotted lines in graphics of Figure 4.10). We aim at finding heuristics on the essential binary information to preserve for lossy compression, in the spirit of the entropy bound for lossless source coding. Delving into the extensions of the above contributions, taking advantage of the data anisotropy or investigating the bit plane encoding may offer novel insights.

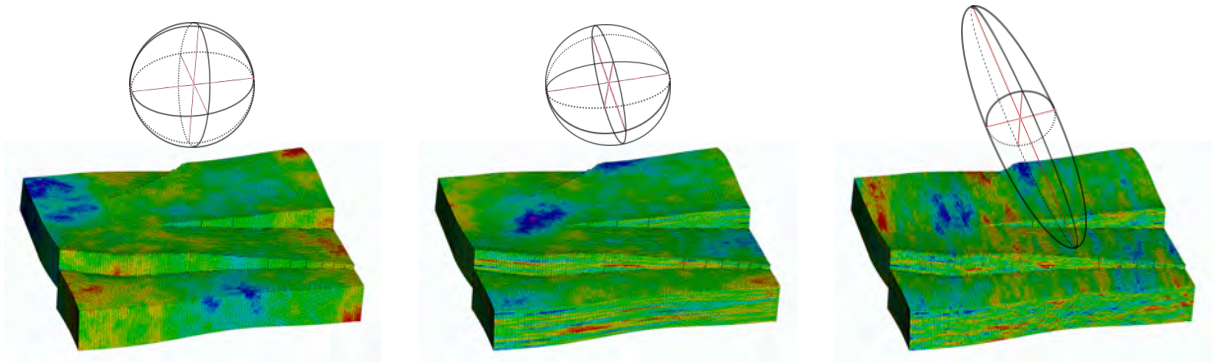
4.3 Deepening the method

We present in this section partial investigations and intuitions that appear to be promising for optimizing the RM compression. The anisotropy of our meshes first inspired us. This feature particularly observable on properties can be studied using a classical statistical tool in geosciences, called variogram. It allows determining preferential orientations in the distribution of petrophysical properties. In the next subsection, we test a packet wavelet decomposition along these orientations, to exploit this feature. Such a method should increase compression performances without degrading the simulation results.

In the second subsection, we focus on the ZT-based encoding stage. The study of its intrinsic parameters provides information on the data structure and could advise on the minimum amount of binary data required for simulation. Their analysis could even be more complete than the objective study performed so far.

4.3.1 Exploiting the anisotropy

For this experimentation, our approach is still evaluated on the continuous properties of the four geological environments, generated on purpose with anisotropic behavior. The distribution of these properties is conditioned by the depositional environment: sediments, initial sedimentary basin topology, sediments transport, diagenesis (Fowler and Yang, 2003) as explained in Subsection 3.1.3. Such parameters are directionally dependent, subject to marine/fluvial current during the deposit and more generally by the gravity, the pressure from the upper layers, or other constraints during all its genesis. The sedimentary structure is stratified, composed of horizontal layers of varying thickness and composition. By definition, this structure is thus spatially anisotropic along $(\vec{i},\vec{j})$, with a potentially different behavior along a depth $(\vec{k})$, as shown in Figure 4.11. Consequently, considering that these properties are isotropic is probably suboptimal for compression.



Isotropic

Isotropic on surface
Anisotropic through the depth

Anisotropic
in three directions

Figure 4.11: Continuous properties modeled from three different variograms. The ranges in the three main directions (red lines) are illustrated by the ellipsoid shape.

Exploiting spatial properties distribution with variograms

A variogram, defined from the Equations 4.5 and 4.6, is a geostatistic tool used to describe the spatial variability/continuity of properties in specific directions. It is defined as the variance of the difference between values at two points separated by a distance h, equivalent to a number of cells.

$$\gamma(x_i, x_j) = \frac{1}{2}\mathrm{Var}(Z(x_j) - Z(x_i))\,. \tag{4.5}$$

$$\gamma(h) = \frac{1}{2N(h)} \sum_{i=1}^{N(h)} (Z(x_i + h) - Z(x_i))^2\,. \tag{4.6}$$

The (uniform) distance separating points x_i, x_j is denoted by h, while $N(h)$ denotes the number of pairs of points (separated by distance h).

Along the three main directions $\vec{i}$, $\vec{j}$, and $\vec{k}$, three 1D variogram curves can be computed. As illustrated in Figure 4.12, the curve is described against its *sill* (asymptotic limit), and its *range* (distance in which the difference of the variogram from the sill becomes negligible).

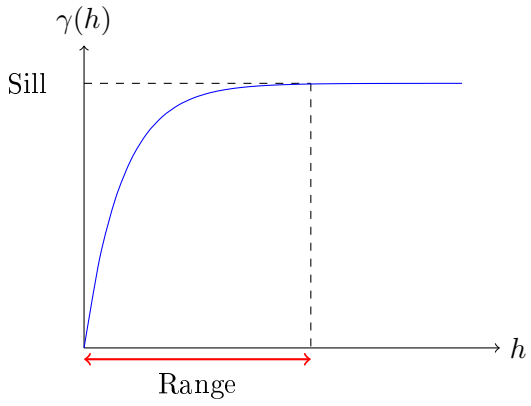


Figure 4.12: Variogram curve in one direction, for an exponential model (among spherical, gaussian, cubic *etc.* models).

Estimating the variogram could figure as a preprocessing to compression, to determine some preferential directions in the data. An anisotropic wavelet decomposition could then be applied to benefit from this feature. This kind of approach has been already proposed in Christophe et al. (2008) for hyperspectral data. This work uses wavelet packets on approximation and detail subbands localized along directions pointed by the study of variograms.

Our geoscientific data also presents anisotropy. Here we focus on the petrophysical properties introduced in Subsection 3.5.3, whose the anisotropy orientation should be along the vertical axis $\vec{k}$ but is not necessarily intuitive, considering the various cell dimensions. The variogram of the porosity in the nearshore$_1$ environment is displayed in Figure 4.13 (the other environments can be found in appendices, page 128). The range distance initially expressed in meters ($4.20 \times 4.20 \times 1$

meters), is converted into number of cells, because of property discretization and average dimensions cells. We observe that the range along $\vec{i}$ and $\vec{j}$ are ten times higher than the range obtained along $\vec{k}$. This means that along $\vec{j}$ for instance, two cells separated by less than 55 other cells are likely to be correlated. Above that distance, a weaker link is expected along the $\vec{k}$ direction. Such a spatial distribution is common and these variograms are comparable to those obtained on the Tarbert formation (from SPE10 model), equal to (14,32,3).

Because the range computed in the $\vec{k}$ direction (depth) is short, we assume that the information changes fast. Classical dyadic decomposition can leave behind information on detail subbands that are still correlated.

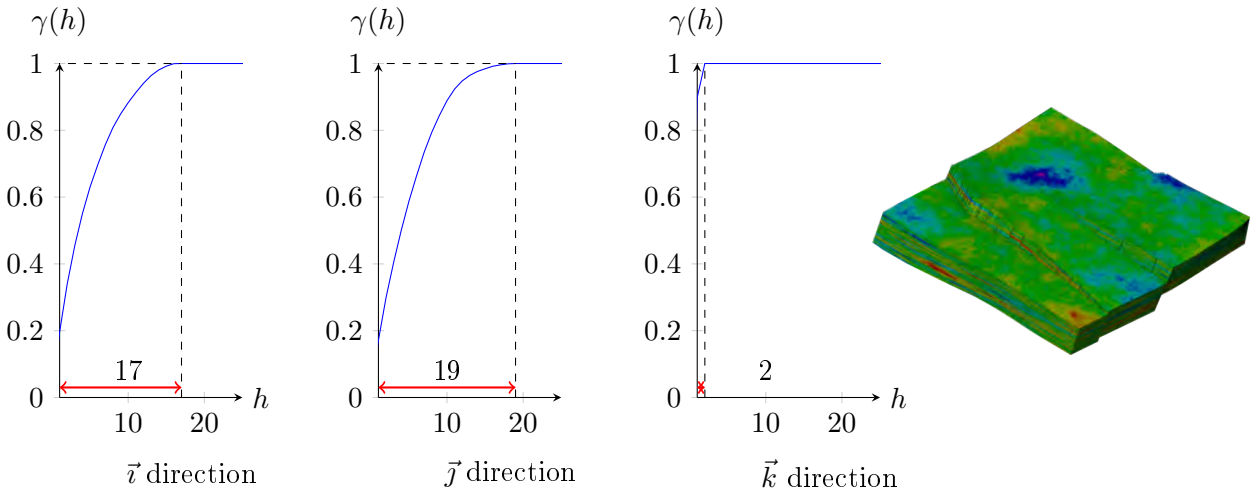
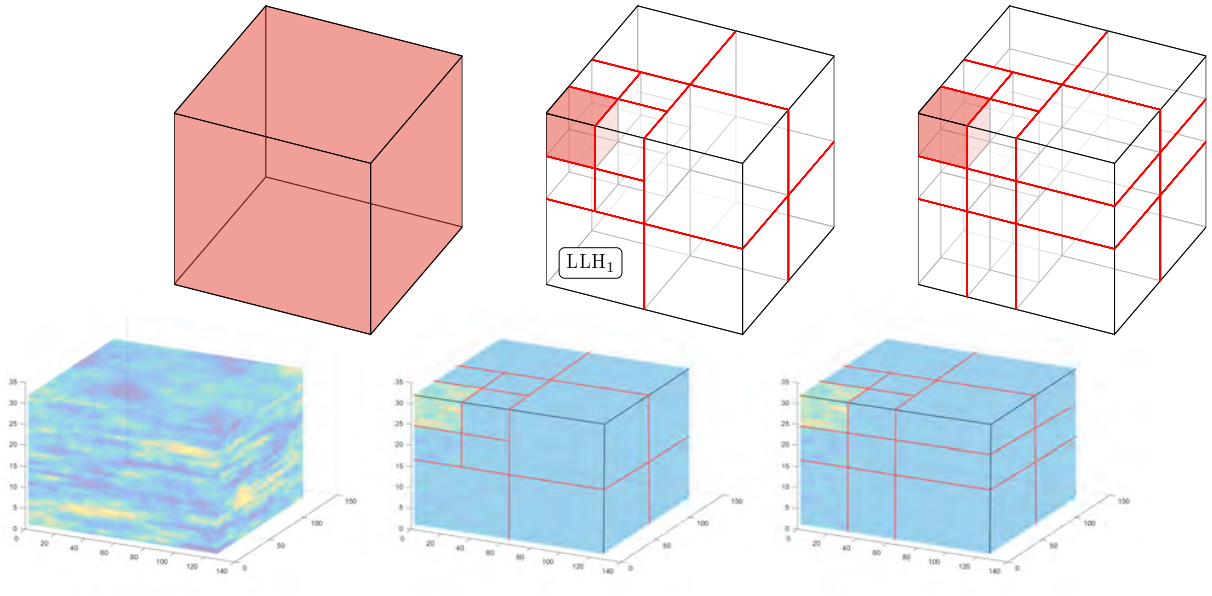


Figure 4.13: 1D variogram curves of the permeability property in the nearshore$_1$ environnement.

Preliminary results with anisotropic decomposition

Figure 4.14 illustrates the two approaches compared in this section. The volume data on the left (preprocessed by $\Lambda_{0,25bits}$) is decomposed with a 3D dyadic wavelet, generating a structure displayed in the middle of the figure. For this experimentation, we used 25 bits for Λ (instead of 34 bits) and two decomposition levels (instead of five) for sake of simplicity on the figure. This method is compared to the wavelet packet decomposition illustrated on the right, more suitable to handle data anisotropy. The previous study of variograms recorded a very short range along $\vec{k}$, showing sudden spatial changes and concentration of information in this specific direction. Despite the first dyadic decomposition, the LLH_1 detail subband is still correlated. We can recognize in this subband an approximation of the original data at a lower resolution, rather than detail data. It should therefore look more like noisy data, and be composed of small values. At this place, the wavelet packet decomposition uses two additional decompositions to better decorrelate information on this subband.

To assess the interest of using a wavelet packet transform to take into account the data anisotropy, we compare the weighted entropy (Shannon, 1948) of the two approaches. The entropy designates



	$\Lambda_{0,25bits}$	2 dyadic wavelet decompositions	2 wavelet packet decompositions
weighted entropy (in bits per value)	18.92	15.62	14.99

Figure 4.14: Dyadic decomposition (middle) vs wavelet packet decomposition along $\vec{k}$ (right). The initial data (left) is the permeability property in nearshore$_1$ environment preprocessed by Λ_0. The weighted entropies for each data is indicated in the bottom part of the figure.

the asymptotic number of bits per symbol required to transmit the minimal quantity of information in a lossless compression perspective. For a data constituted by $\mathcal{C}$ discrete values, each value c has a probability of occurrence equal to P_c. The entropy, in bits per value, is defined by

$$H = -\sum_{c=1}^{\mathcal{C}} P_c \log_2(P_c).$$

In case of decomposed data, the weighted entropy is generally used. It consists in weighting the entropy of each subband by the ratio between its number of cells and the total number of cells:

$$wH = \sum_{k=1}^{N} \frac{\text{nb cells in subband } k}{\text{nb cells in data}} H(k).$$

Computed on the entire data preprocessed by $\Lambda_{0,25bits}$, the weighted entropy is close to 19 bits per value, whatever the environment (all the results are in appendices, page 128). Considering the property in the nearshore$_1$ environment, the classical $3D$ dyadic decomposition reduces the weighted entropy from 18.92 to 15.62 bits per value. With the wavelet packet decomposition, the weighted entropy reaches 14.99 bits per value, which is thus lower.

As demonstrated previously, our evaluation method introduced in Section 4.2 is complete and relevant to assess compression in a simulation workflow. We therefore use it in the same way in this section to assess the anisotropic wavelet decomposition. For this experimentation, we made again five decomposition levels and 34 bits, to be compared with former results. Results displayed on Figure 4.15 in black thus complement the graphs of Figure 4.10.

So far, the performance of our alternative method was close to the concurrence. The anisotropic approach, based on Λ_0, a wavelet packet decomposition and an adapted ZT structure (Christophe (2006)) surpasses the concurrence results. This experimentation confirms the positive effect of Λ_0 compared to Λ_1 : solid lines are always above dashed lines. In addition, the objective thresholds determined earlier, correlated to subjective evaluation, are still valid (dotted blue lines). In the fluvial environment, the best result was held by SZ (point wise relative option). The lower data precision able to satisfy the simulation workflow is generated with a budget of 8.92 bits, and assessed by a Λ-SNR equal to 53.77dB. For a lower bit budget of 8.83 bits, the new anisotropic method (black solid line) generates a data at higher objective quality, equal to 56.77 while maintaining "identical" simulation results.

In short, wavelet packet decompositions show promising performance, and would require in-depth experimentations.

4.3.2 Thresholding at necessary precision

The objective of this last experimentation is to determine the precision required for suitable simulation data. Up to now, our results designate thresholds by using adapted objective metrics (blue dotted lines). Nevertheless, the threshold value varies according to the data type. Therefore our research pursues a quest to address the minimum information, regardless of the handled data.

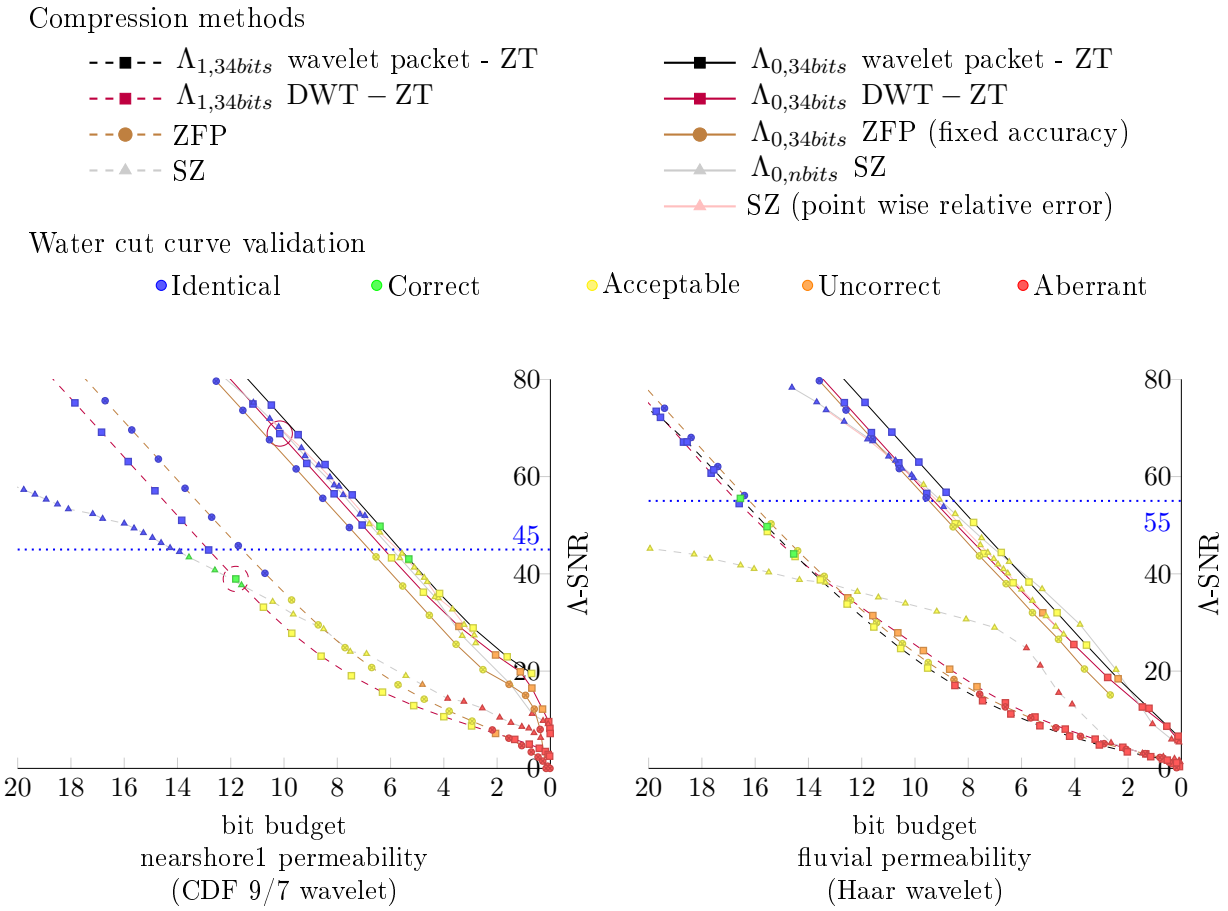


Figure 4.15: Permeability from nearshore$_1$ (left) end fluvial (right) environments generated at refinable precision decreasing the bit budget (a mark per precision) using our method (wavelet packet or dyadic decomposition), SZ and ZFP (using or not compandor), objectively evaluated by Λ-SNR. Subjective appreciation of simulation results is represented by a color code.

We focus henceforth on encoding parameters, in particular on the *ZT activity* measured for each bit plane and below defined. The notion of *bits per bit plane* has been particularly studied by Wang et al. (2019) to describe the ZFP activity, while others have detailed the encoding process of SZ.

The standard ZT coder (Shapiro, 1993) uses four symbols to encode wavelet transformed data (POS, NEG, ZTR, IZ). They determine the status of a coefficient for a particular bit plane considering its previous state in the higher bit plane. As explained in Subsection 3.5.2, the coefficients are successively visited and compared to a decreasing threshold equal to $2^{\text{bit plane}}$. If the value of the coefficient is higher than the threshold, the coefficient is considered significant, and is included in the list of significant coefficients. From this bit plane, the coefficient is encoded using POS and NEG symbols. On the contrary, if the coefficient is lower than the threshold, it means that all its bits considered so far were zero. Depending on its location in the decomposed structure, a coefficient figures as the ZT root (encoded with a ZTR symbol) or belongs to a ZT (by consequence it is implicitly represented by a root element) while IZ is for "isolated zero".

Consequently, the state of a coefficient changes continuously using lower thresholds. From unexpressed, it is potentially identified as a ZTR or IZ and finally becomes significant. We define the *activity* as the number of coefficient state changes for a bit plane, with nSIG the number of significant symbols (POS, NEG), and nZ the number of zero symbols (ZTR, IZ) at the i^{th} bit plane. The activity is relative to the total number of coefficients noted $\mathcal{C}$, and defined by:

$$\text{ZT activity}(i) = \frac{\text{nSIG}(i) - \text{nSIG}(i+1) + \text{nZ}(i) - \text{nZ}(i+1)}{\mathcal{C}} \tag{4.7}$$

Figure 4.16 shows the curve of *ZT activity* according to the bit plane for the permeability in the nearshore$_1$ environment, preprocessed by $\Lambda_{0,34bits}$ and decomposed by the CDF 9/7 wavelet. This curve describes the progressive encoding process of binary data from the MSB to the LSB. Consequently, this reverses the quality progression (Λ-SNR, bit budget) drawn so far, that showed the precision decreasing from left to right using less bit budget along the x-axis. Now, reading along the same axis involves using more and more binary planes, thus increasing the precision of the reconstructed data.

We interpret the *ZT activity* as a clue on the encoding evolution and allows identifying different phases. At beginning, using a partial number of MSB_M planes, ZT are few but constitute large structures (comprising a large number of zero coefficients). Consequently the starting *activity* is low because only few nZ and nSIG are encoded. Progressively, the number of ZTs tends to increase by splitting precedent large ZTs into smaller ones, while ever more coefficients become significant. For the example of Figure 4.16, the *ZT activity* takes off from the 36^{th} bit plane and peaks at the 31^{th} bit plane. Then, the *ZT activity* decreases to zero as well as the number of ZTs, since all coefficients become significant using an ever lower threshold. From the 22^{th} bit plane, we observe that the ZTs vanish. However, the ZT can continue to encode bit planes, even though they no longer have a structure. Our postulate is that, beyond this bit plane, information consists of incoherent data that does not contribute to the increasing of the precision for the reconstructed RM.

Besides, if considering the renderings, as soon as the *activity* returns to zero, *i.e.*, once the 22^{th} bit plane is processed, the (de)compressed data is already very close to the original data visually (indexed by "8" in the Figure). The decoding of the subsequent bit planes will only add very light

visual details. If considering now the simulation accuracy, as soon as the *activity* returns to zero, our simulation results are already "identical". This can be seen thanks to the color of the marks on the *activity* curves that allude to the subjective evaluation, as previously done in Subsection 4.2.3. Finally, it would seem that we learn more about the data precision required to have a good visual quality and also an unbiased simulation outcomes by interpreting *ZT activity* than by analyzing objective metrics such as the curve Λ-SNR, displayed at the bottom of the figure.

Figure 4.17 shows the curve of *ZT activity* according to the bit plane for the permeability in the nearshore$_1$ environment, but this time preprocessed by $\Lambda_{1,34bits}$. The curve is quite similar than with $\Lambda_{0,34bits}$. A peak is observable between the 36 and 18^{th} bit plane, although less pronounced. But the conclusion is the same: as soon as the *activity* returns to zero, the visual quality is good, and the simulation accuracy is already considered as "identical". It confirms the fact the *ZT activity* seems to be a relevant information to assess the quantity of data required for simulation, whatever the compandor features.

To summarize, in this last section, we showed that the compression workflow could be probably improved, first by taking into account the anisotropy of some continuous properties. It could be simply done for instance by changing the dyadic decomposition by a wavelet packet decomposition, which enhances the decorrelation of some subbands neglected by an isotropic approach. Second we showed that the decrease of *ZT activity* during (de)compression seems to indicate the minimal data quantity required for obtaining accurate simulation outcomes even with a limited precision (as for displaying very nice reconstructions, with negligible visual losses). Finally, we also showed that, even if the bit planes subsequent to the peak of *activity* do not further improve the quality of the decompressed data, the curves of the objective measurements (Λ-SNR) continue to increase linearly, demonstrating an "illusory" precision. Actually, the linearity of this curve does not allow us to distinguish the significant data from the rest considered as noise by (Natarajan, 1993).

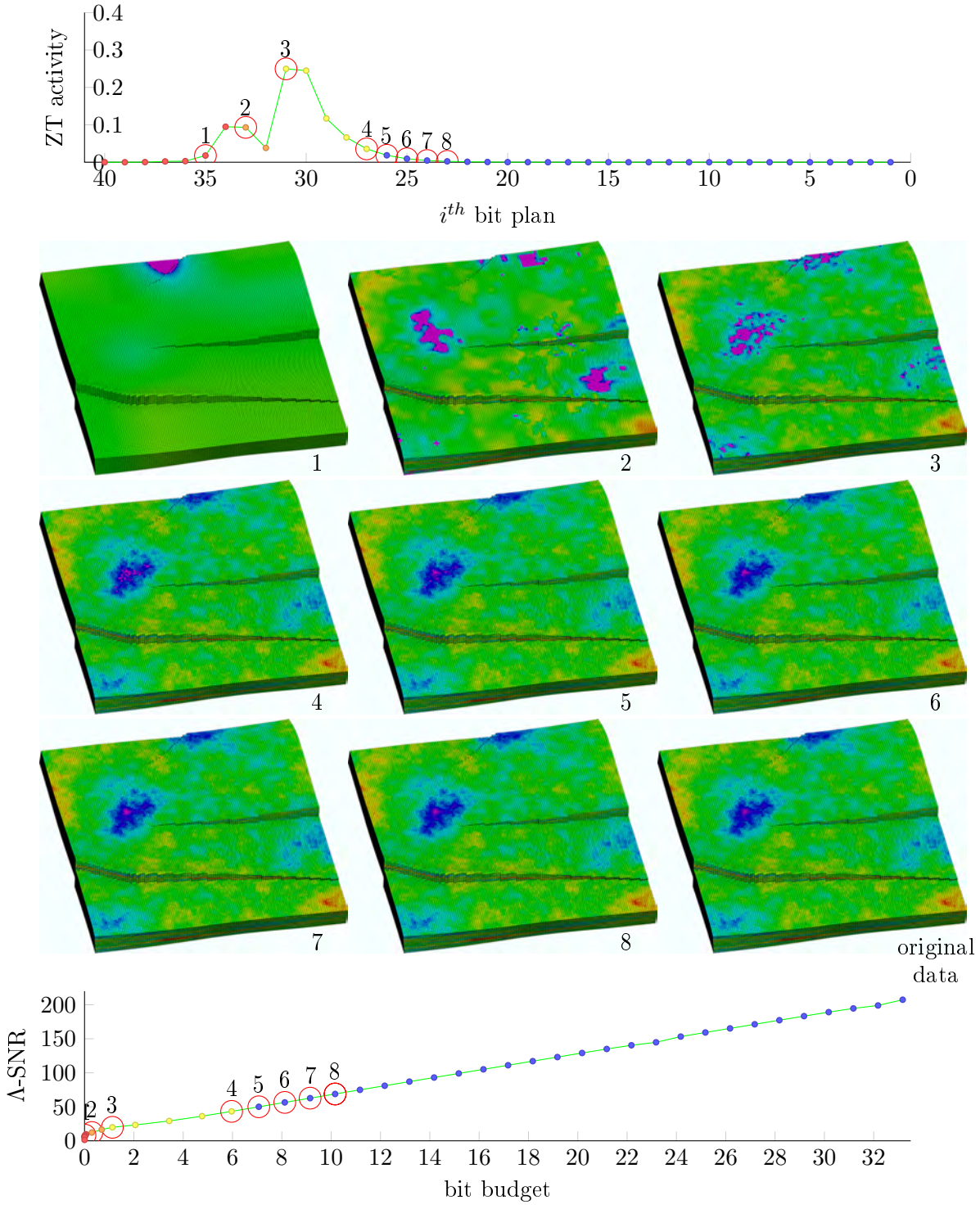


Figure 4.16: Correlation between ZT activity (top-curve) and objective evaluation by Λ-SNR (bottom-curve) of permeability from nearshore_1 environment generated at refinable precision by increasing bit planes number for reconstruction while our compression alternative using Λ_0.

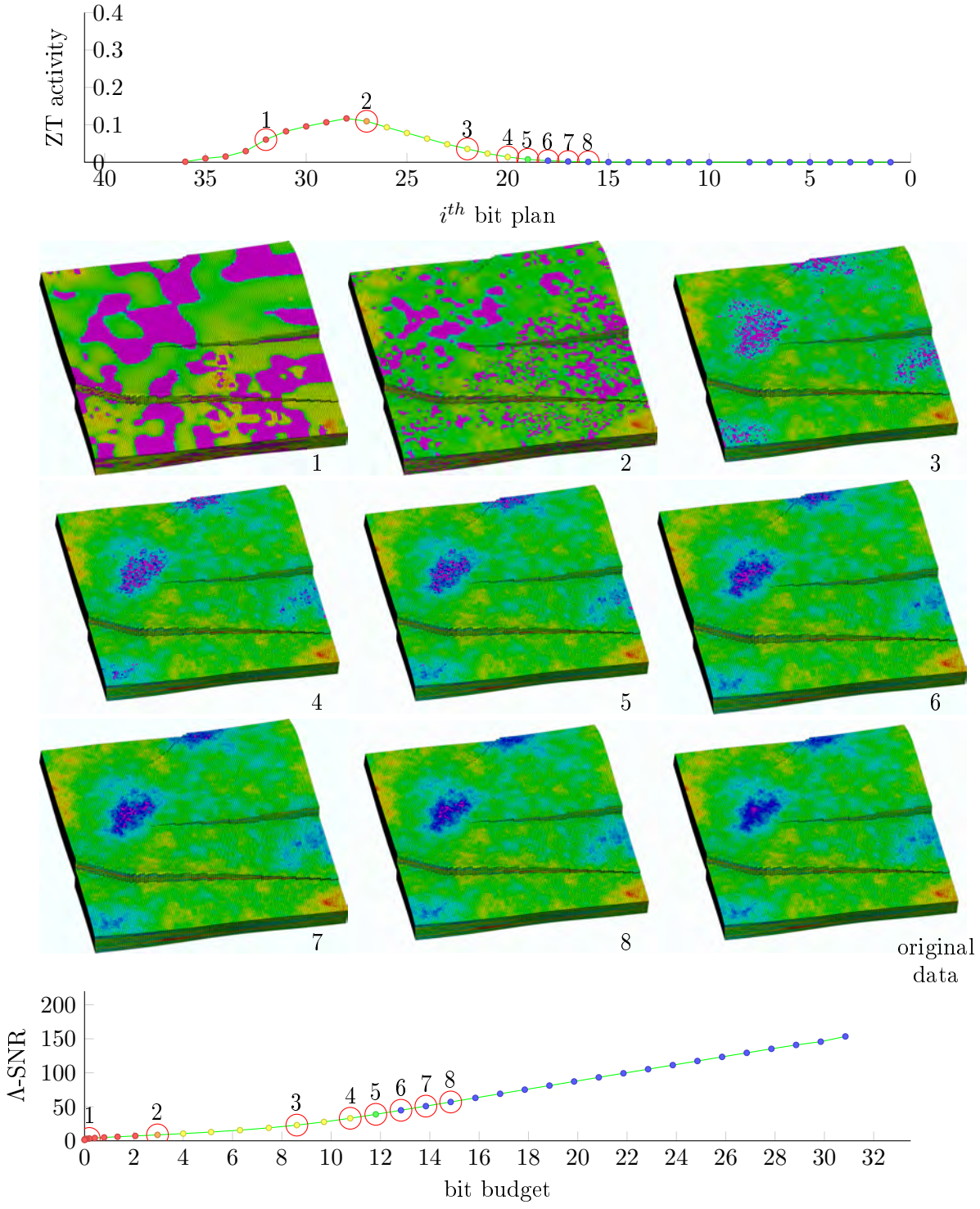


Figure 4.17: Correlation between ZT activity (top-curve) and objective evaluation by Λ-SNR (bottom-curve) of permeability from nearshore$_1$ environment generated at refinable precision by increasing bit planes number for reconstruction while our compression alternative using Λ_1.

CHAPTER 5

Conclusions & perspectives

5.1 Conclusions

This work evaluated the relevance of compression for reservoir meshes, from visualization to simulation through lossless and lossy storage, along four chapters. The first chapter provided context on the growing concern of data handling in data-intensive science. In a second chapter, we focused on geosciences and volume meshes used to model objects and simulate geological phenomena and reservoir behaviors. The third chapter reviewed methods employed to deal with huge datasets in various scientific fields. We then introduced a comprehensive multiscale approach for the different components of geological volume meshes. Its evaluation for visualization and compression shows the benefits of combining embedded scales with both generic and refinable data encoding techniques. The impact of refining spatial resolution and numerical precision on a complete simulation workflow is finally assessed in Chapter 4 on the LUNDI reservoir model. We thereafter provide more details on our contributions.

Compression in the domain of multimedia is very commonly used to reduce data quantity and improve the distribution of sound, images and videos. Compression standards (jpeg, mp3) have gradually invaded our daily digital lives, without end users being really aware of their benefits and implications. Compressing digital data often involves the concatenation of regular components following a logical scheme. The singularity of a compression algorithm often lies in the nature of its components, chosen to meet specific needs: execution speed, compression ratio, quality. It is hardly optimal on all fronts. Hence, it is therefore essential to identify user needs, to find a balance between theoretical advances and practical concessions. This allows to propose adaptable and perfectible models for various data sources.

Contrary to multimedia, data formats in sciences is not well standardized, as they are not in-

tended, originally, to be exchanged amid a wide audience. However, their quantity exploded as well in the last decade, in a quest for better reality representation, leading to increased model precision and resolution. Technological advances on computational resources and facilities (high-performance computing, gigantic data centers) allows the processing of ever more detailed data. Yet, this system is gradually touching its limits, and bottlenecks are appearing in data storage, transmission, or computing. To address these growing issues, researchers have being re-examining compression for scientific data, in a variety of application fields with very different data nature and dimensions. In geosciences, volume meshes are complex data made of heterogeneous elements (from structure to properties). Their reduction — to better handle their growing dimensions and therefore reduce the computation time of reservoir flow simulations — is already a research subject. Still, the variety of models and the assessment of quality at different steps of a simulation workflow still requires attention.

We demonstrated in the third chapter that compression can significantly reduce the quantity of scientific data by studying diverse meshes and different compression tools, with approaches ranging from lossless to lossy. In the spirit of upscaling methods deployed in geosciences, we base our generic hexahedral mesh representation on HexaShrink, a multiscale decomposition offering dyadic intermediate resolutions. It is principally non-expansive: lower resolutions are embedded in a single structure that does not increase the number of binary objects. It is based on different kinds of discrete wavelet decompositions, adapted to each mesh component. We take a special care in the visual preservation of discontinuities, like faults. The relevance of this scheme for visualization is verified on eight meshes, in a comparison with lower-resolution grids obtained from known geomodellers.

HexaShrink allows a sparser representation of regular data variations. Combined with several generic encoders, this hierarchical decomposition is thereafter proved efficient for storing meshes in a lossless manner. Their size can be decreased by a factor of two to ten, while faithfully preserving their exact information. A detailed analysis for each mesh component shows huge differences in compression ratios. Contrary to structure data, continuous properties were hardly compressible. This is corroborated by the literature, and related to the dynamic range of petrophysical data with classically-used scientific floating-point formats.

This observation encouraged us to pursue our work on compressing continuous properties with an evolved progressive coder, named zerotree. It aims at better handling multiscale decompositions by exploiting remaining redundancies in the transformed data. Besides being adapted to spatial refinement across resolutions, it also parses data in a binary-depth order. Contrary to the lossless methods used before, data can be generated at refinable precision, allowing a progressively lossy compression, from the most to the least significant bits.

The quality of reconstructed data was evaluated on a wide range of bit-per-cell values, using objective metrics and visual clues. A focus was laid on continuous properties of LUNDI, a reservoir model designed for the purpose of a second comparative benchmark. We demonstrated that a wavelet transform associated to zerotree coding was again competitive in performance with respect to generic encoders. It saved at best about ten bits per value at single floating-point precision (32 bits). It however requires a strict control of degradation, to be used with confidence in simulation.

Indeed, in the fourth chapter, we demonstrated that compression may used very little negative impact on simulation, if properly guided. It is analyzed on the LUNDI reservoir mesh across a cus-

tomizable base of simulation parameters. Motivated by the difficulties to compress permeability data, we complemented the wavelet/zerotree compression with compandor, a scalar data transformation inspired from physical principles. We also proposed a novel family of compandor-modified objective metrics. Although simple, they proved well-suited to objectively measure the subjective quality of well production evaluated on compressed meshes.

To be able to conduct the above proof-of-concept for a compression scheme with refinable resolution and precision, the HexaShrink decomposition was chosen isotropic. Expecting increased performance with the knowledge of reservoir data orientation, and guidance on the actual precision needed for accurate simulation, we finally explored two complementary directions. The first one uses anistropy information obtained from variograms, and suggested that sparser decompositions (hence better compression) could be achieved with wavelet packets instead of wavelets. The second one consists in investigating the binary activity of bit depths in zerotree encoding. Looking at the binary precision of the data, we may detect when encoding has already attained a limit of structured information that can be used to compress data. Below that limit, the binary bit planes add little to no information. They thus carry a noise-like content, and could be discarded as unuseful.

These preliminary results however remain solid intuitions, showing promises to be confirmed in future works.

5.2 Perspectives

We proposed a comprehensive assessment for the validation of compression methods to be used in a simulation workflow. It included lundi, a versatile reservoir model, a combination of subjective evaluations and novel objective metrics. The multiscale HexaShrink representation is usable throughout a complete simulation workflow (visualization, lossless and lossy compression, model simulation). It provided overall positive performance with respect to state-of-the-art tools. However, while solid, this work could be complemented in the following directions.

The dataset for comparative evaluation could be extended. The lundi mesh could be filled with different property distributions, additional topological discontinuities and especially increased in volume[1]. This motivates our goal to share those datasets with the simulation community, to allow other independent benchmarks to be run.

By studied other volume meshes and different property natures, we could further investigate the use of companding as a useless preprocessing step, and the associated compandor metrics for objective/subjective assessment.

As geoscience meshes are composite, a final compressed bitstream for all the objects should be obtained. This requires to implement progressive coding on every component: activity, corners, pillars, discrete properties, etc. Complexity may arise from the interdependence of the latter: depending on a rock type, and its thickness, its continuous altitude might be quantized at lower precision, without affecting notably the main simulation features. Being able to exploit — jointly — their impact has potential in improving compression ratios.

We have obtained limited results regarding the acceleration of simulation: only the first lower resolutions from HexaShrink can be used for upscaling. We may expect improvements by embracing anisotropy, and notably dissociate the decomposition along depth. This would increase the

[1] We use a limited size of $128 \times 128 \times 32$ for efficiency reasons, to be able to performed a large quantity of simulations.

complexity of the global coding of all mesh components.

Finally, a major line of research would reside at the interface of simulation and compression. Simulation codes use complicated refinement schemes, potentially on multiresolution grids. They can be sensitive to the data precision, hence the use of extended floating-point formats. They are not, to the best of our knowledge, devised to use the prioritization of information that can be afforded by compression tools: prominent information at specific resolutions, precision that can be refined upon numerical accuracy bounds, etc. Such a blending of compression pipelines and simulation workflows could yield gains in both data storage and computational time.

List of Figures

List of Tables

Appendices

Subsection 3.4.3: In-depth analysis of the coding performance#

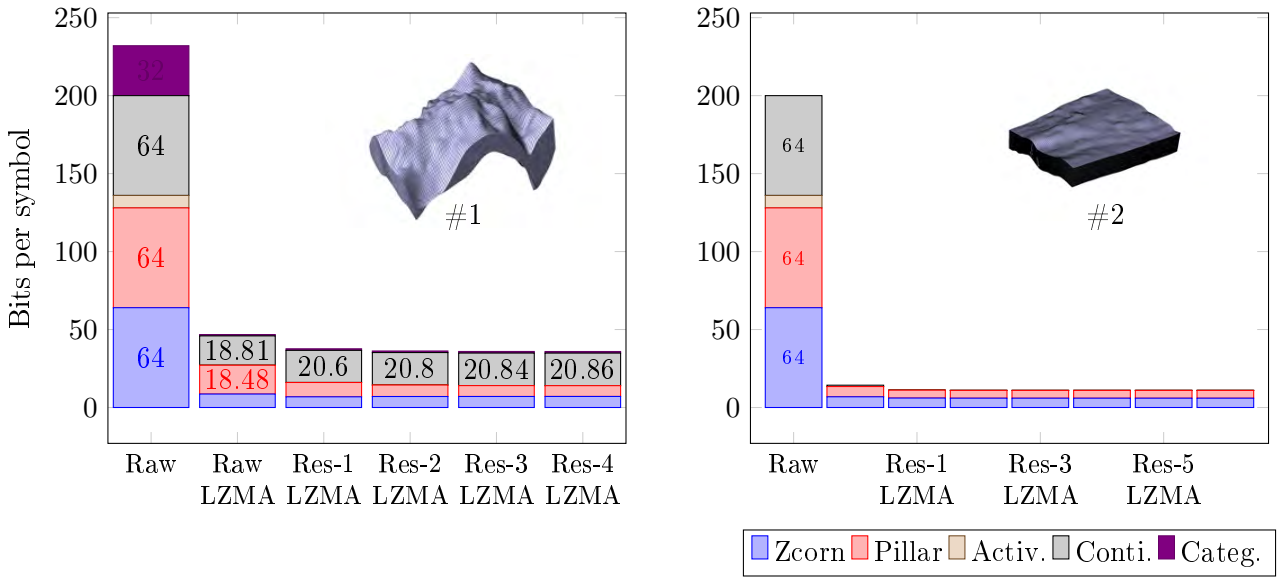


Figure 5.1: Binary cost of each component for mesh#1 and mesh#2 in function of the number of successive HexaShrink decompositions. Complementary results from Section 3.4. If the mesh contains two continuous properties, only the results for porosity property is presented.

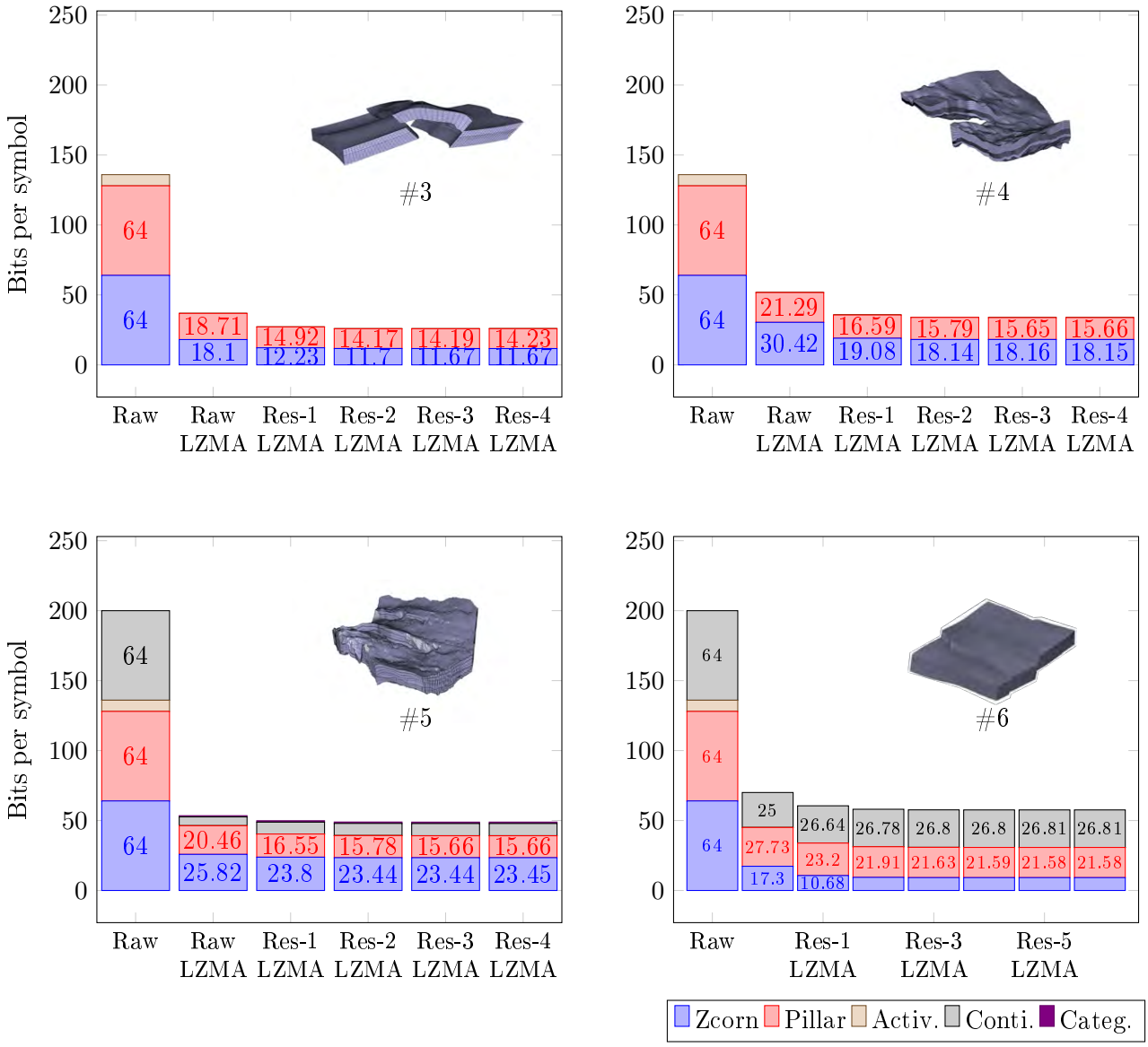


Figure 5.2: Binary cost of each component for mesh#3, mesh#4, mesh#5 and mesh#6 in function of the number of successive HexaShrink decompositions. Complementary results from Section 3.4. If the mesh contains two continuous properties, only the results for porosity property is presented.

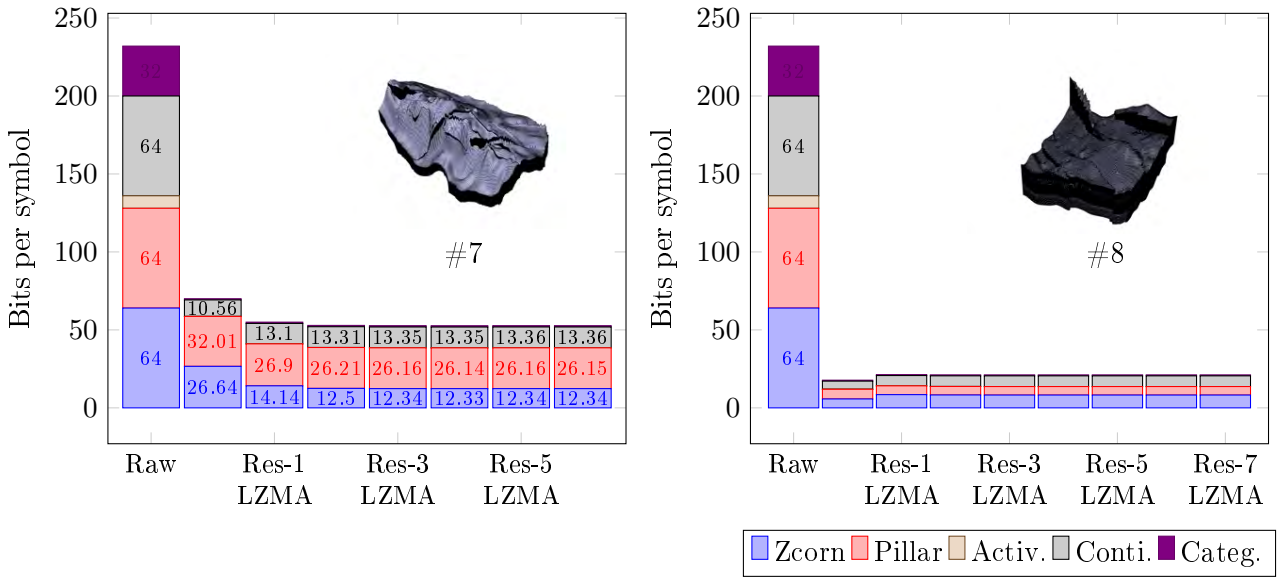


Figure 5.3: Binary cost of each component for mesh#7 and mesh#8 in function of the number of successive HexaShrink decompositions. Complementary results from Section 3.4. If the mesh contains two continuous properties, only the results for porosity property is presented.

Subsection 4.2.1: Subjective evaluation of simulation results#

Fives areas around WC are delineated, as illustrated by Figure 4.3, to define a containing area for $\widehat{\text{WC}}$. From the closer to the most distant, we define: identical - correct - acceptable - incorrect - aberrant areas.

To limit different areas, functions (Equations 5.2 & 5.1) are generated by editing WC reference using specific error parameters. Acceptable error varies according to the category type, and exploitation moment. It tends to progressively increase across time, such as interest of reservoir engineer.

The temporal evolution is detailed on Figure 5.4, by dividing the exploitation duration (almost one year) into three phases, separated by two red dashed lines, noted d_1 and d_2. First line set at water breakthrough (inflection point of WC), while second line points water saturation at 0.2.

During the first phase, extracted fluid does not contain any water. Therefore water saturation is zero or close to zero. There, only tiny constant error E_0 is tolerated by reservoir engineer. During the middle phases, water saturation sharply increases to progressively reach a plateau during last phase. Acceptable errors, respectively noted E_1 and E_2 temporally grows, by cumulation of e_i and a delay for two last periods, as reported in the Table 5.1.

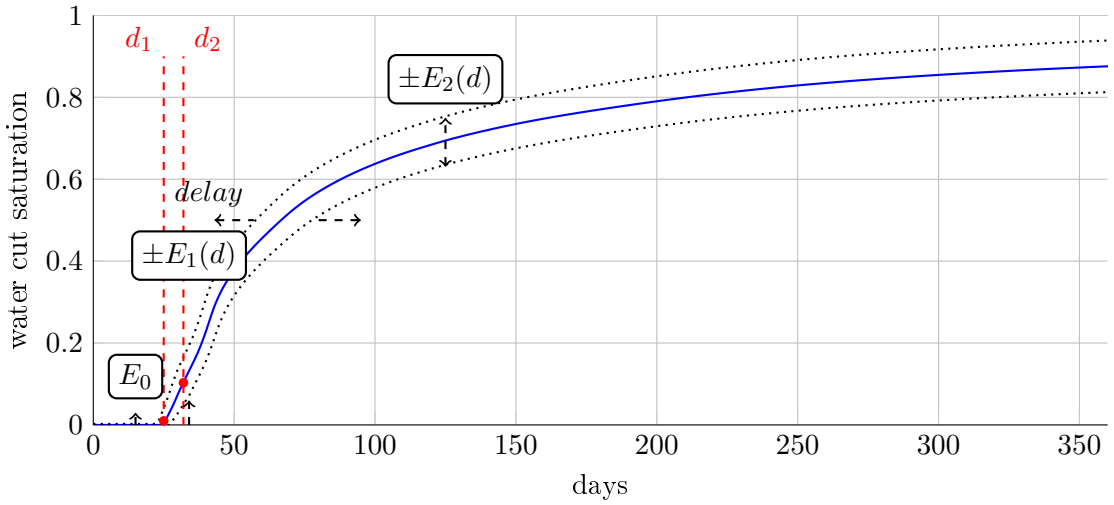


Figure 5.4: Water cut curve in blue obtained on the original data: near shore environement, serves as reference. The water breakthrough is spotted by first red point.

$$\mathrm{WC}_{sup}(d) = \begin{cases} \mathrm{WC}_{ref}(d) + E_0 & ,d < d_1 \\ \mathrm{WC}_{ref}(d + delay) + E_1(d + delay) & ,d \in [d_1,\ d_2] \\ \mathrm{WC}_{ref}(d + delay) + E_2(d + delay) & ,d > d_2 \end{cases} \tag{5.1}$$

$$\mathrm{WC}_{inf}(d) = \begin{cases} \mathrm{WC}_{ref}(d) - E_0 & ,d < d1 \\ \mathrm{WC}_{ref}(d - delay) - E_1(d - delay) & ,d \in [d_1,\ d_2] \\ \mathrm{WC}_{ref}(d - delay) - E_2(d - delay) & ,d > d_2 \end{cases} \tag{5.2}$$

$$with \quad \begin{cases} E_0 = e_0 \\ E_1(d) = e_0 + e_1 \frac{\mathrm{WC}_{ref}(d) - \mathrm{WC}_{ref}(d_1)}{\mathrm{WC}_{ref}(d_2) - \mathrm{WC}_{ref}(d_1)} \\ E_2(d) = e_0 + e_1 + e_2 \frac{\mathrm{WC}_{ref}(d) - \mathrm{WC}_{ref}(d_2)}{\mathrm{WC}_{ref}(d_{final}) - \mathrm{WC}_{ref}(d_2)} \end{cases} \tag{5.3}$$

	Identical	Correct	Acceptable	Uncorrect	Aberrant
e_0	0.001	0.0025	0.025	0.025	
e_1	0.001	0.005	0.02	0.03	Remaining
e_2	0.001	0.005	0.02	0.3	Area
delay	0 day	0 day	1 day	2 days	

Table 5.1: Error parameters

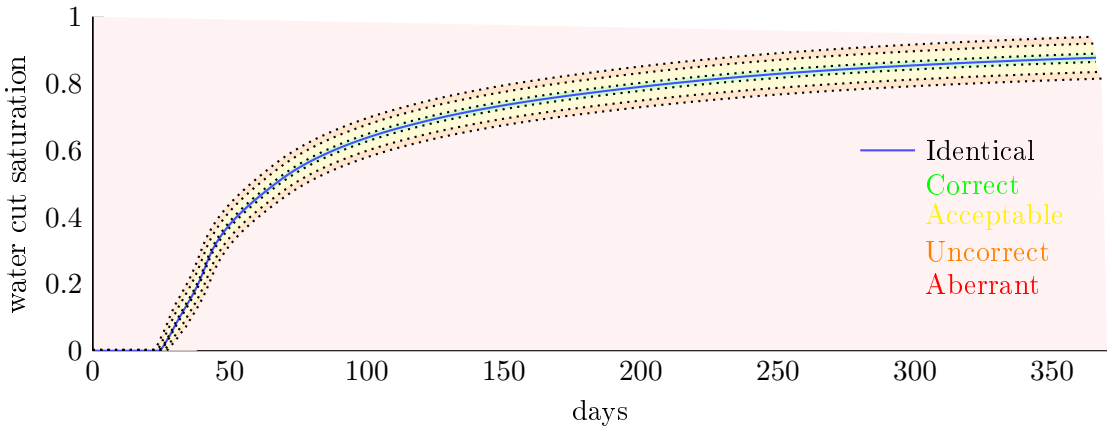


Figure 5.5: Subjective evaluation of simulation results, using region boundaries around the WC (around water cut curve curve from nearshore$_1$ environment) to define a qualitative ranking: 'Identical' (in blue), 'Correct' (in green), 'Acceptable' (in yellow), 'Uncorrect' (in orange) and 'Aberrant' (in red).

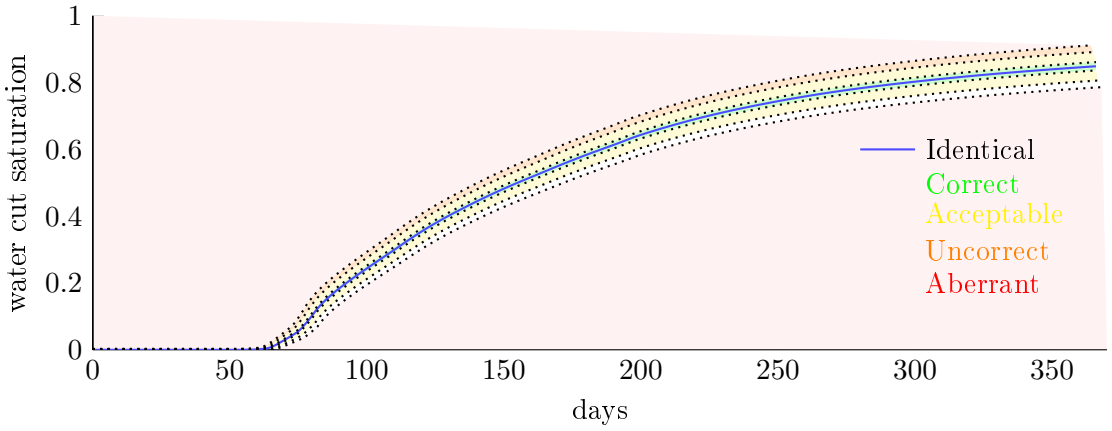


Figure 5.6: Subjective evaluation of simulation results, using region boundaries around the water cut curve (around WC curve from nearshore$_a$ environment) to define a qualitative ranking: 'Identical' (in blue), 'Correct' (in green), 'Acceptable' (in yellow), 'Uncorrect' (in orange) and 'Aberrant' (in red).

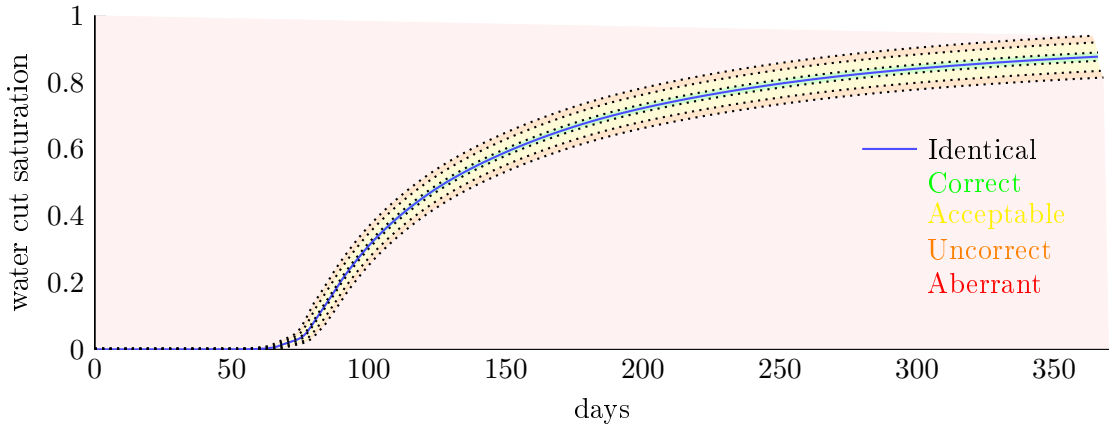


Figure 5.7: Subjective evaluation of simulation results, using region boundaries around the water cut curve (around WC curve from fluvial environment) to define a qualitative ranking: 'Identical' (in blue), 'Correct' (in green), 'Acceptable' (in yellow), 'Uncorrect' (in orange) and 'Aberrant' (in red).

Subsection 4.2.2: HexaShrink evaluation in upscaling/upgridding

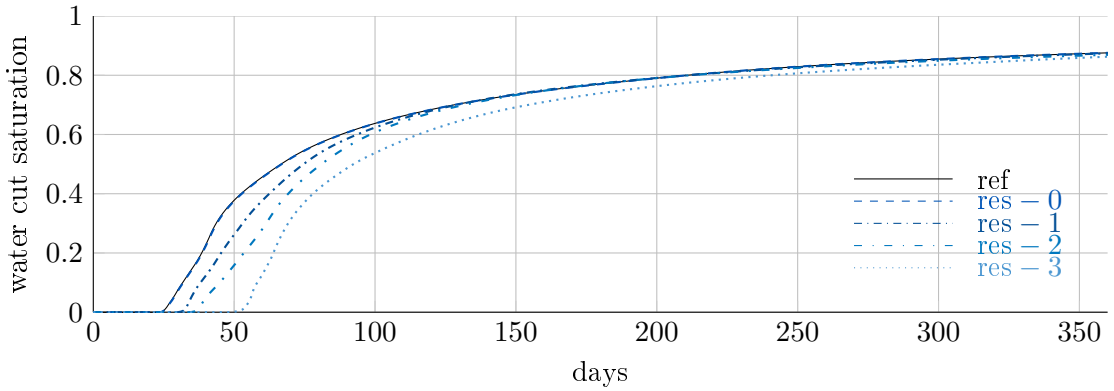


Figure 5.8: Overlay of water cut curve simulated on fine grid LUNDI mesh, (nearshore_1 environment) and $\widehat{\text{WC}}$ simulated on its lower resolutions (up to res-3) generated by HexaShrink.

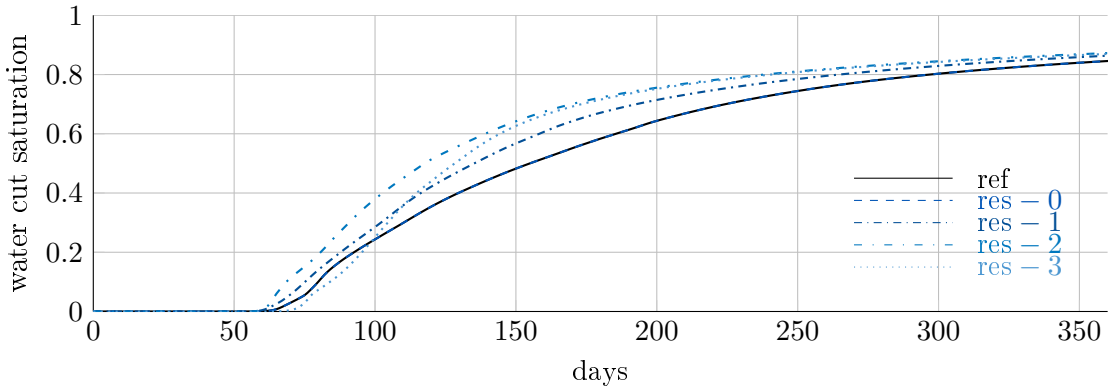


Figure 5.9: Overlay of water cut curve simulated on fine grid LUNDI mesh, (nearshore_a environment) and $\widehat{\text{WC}}$ simulated on its lower resolutions (up to res-3) generated by HEXASHRINK.

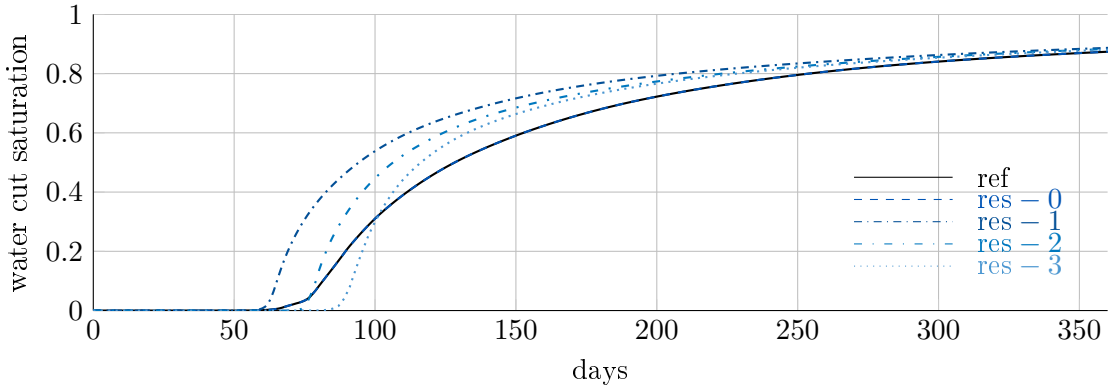


Figure 5.10: Overlay of water cut curve simulated on fine grid LUNDI mesh, (fluvial environment) and $\widehat{\text{WC}}$ simulated on its lower resolutions (up to res-3) generated by HEXASHRINK.

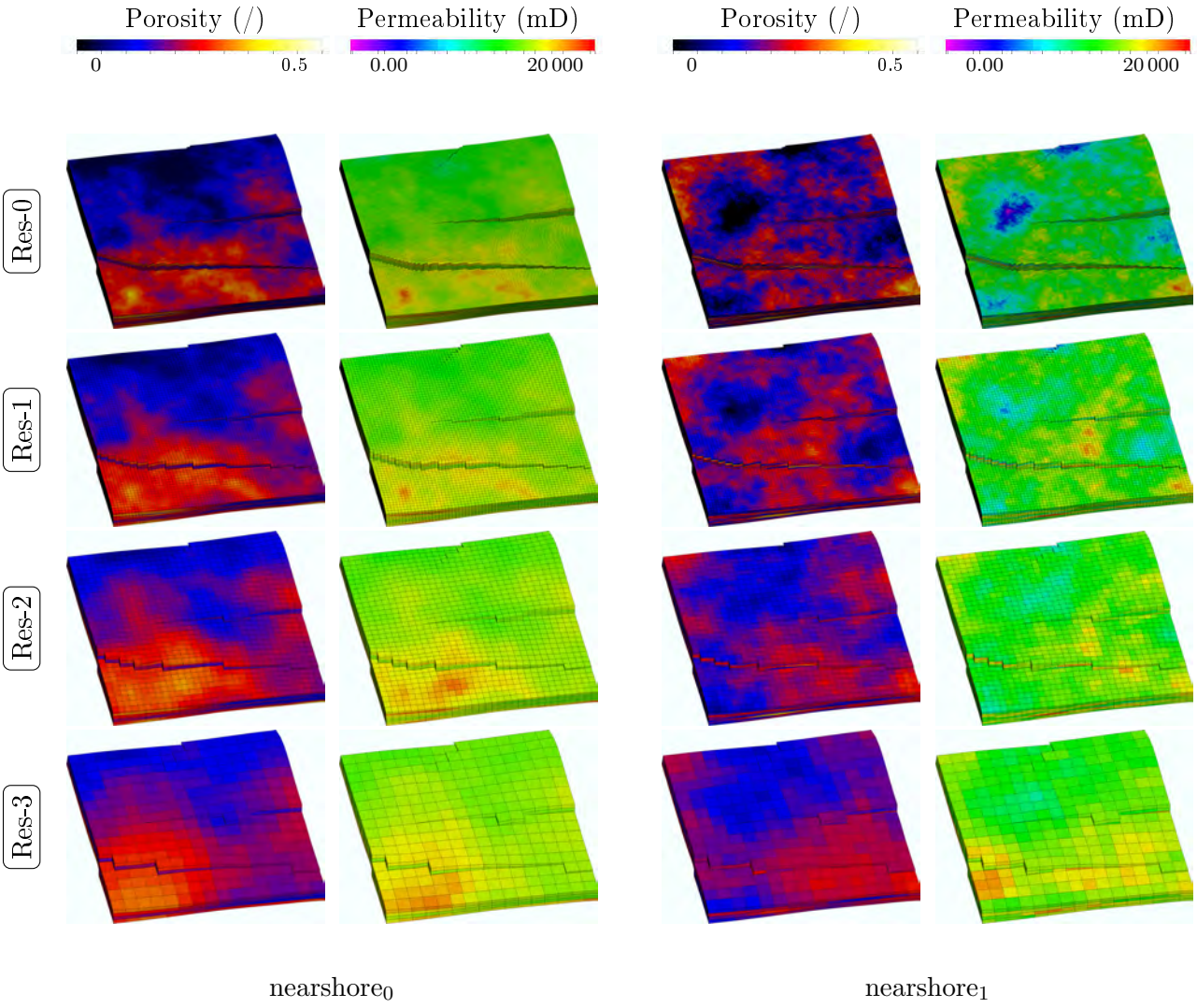


Figure 5.11: Visualization of the LUNDI lower resolution from HEXASHRINK filled by continuous properties, with porosity on upper part and permeability on lower part, for the four distinct geological environments: "nearshore$_0$ " (left), "nearshore$_1$ " (right).

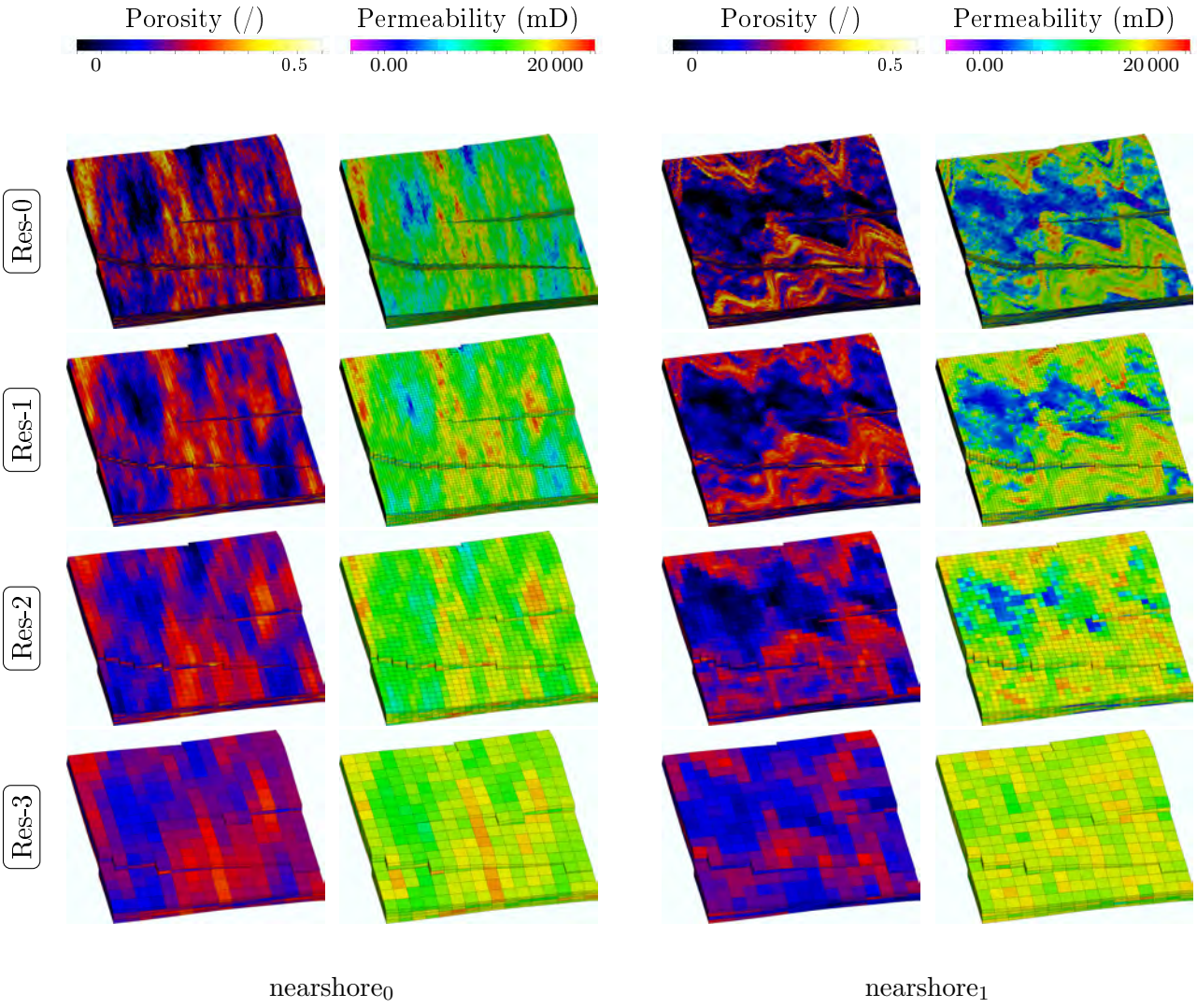


Figure 5.12: Visualization of the LUNDI lower resolution from HEXASHRINK filled by continuous properties, with porosity on upper part and permeability on lower part, for the four distinct geological environments: "nearshore$_a$ " (left), "fluvial " (right).

Subsection 4.3.1: Exploiting the anisotropy

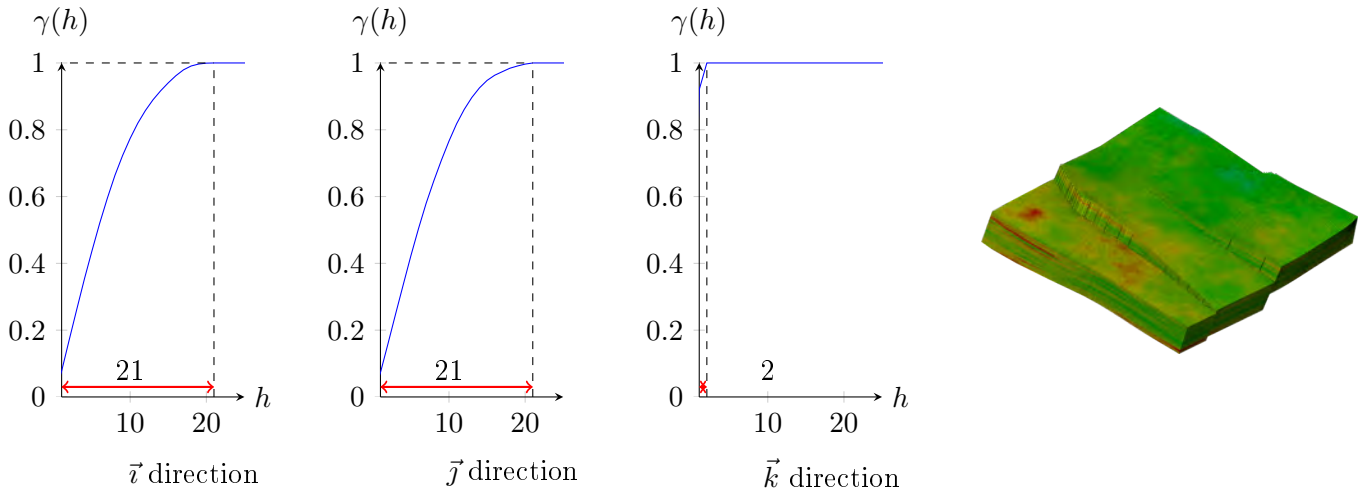


Figure 5.13: Three variograms of the mesh directions computed on continuous property on nearshore_1 environment.

	Weighted entropy		
	Volume data	Dyadic wavelet transform	packets wavelet transform
nearshore_0	18.48	15.58	14.97
nearshore_1	18.92	15.61	14.99
nearshore_a	18.92	15.61	14.99
fluvial	18.81	15.62	15.00

Table 5.2: Weighted entropies computed on permeability property of four environments decomposed by dyadic wavelets or wavelet packets.

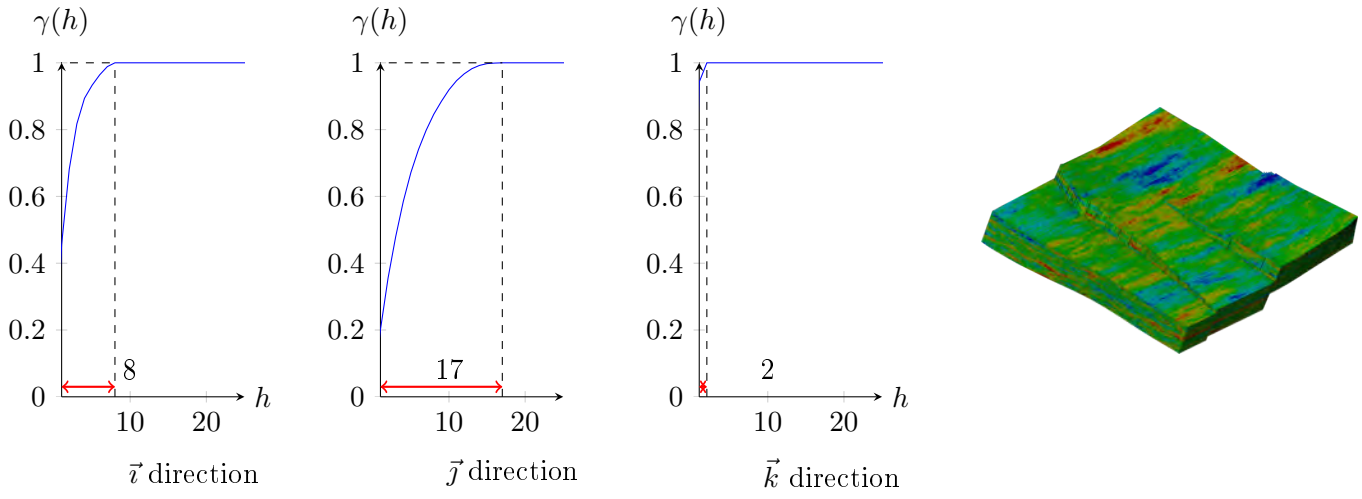


Figure 5.14: Three variograms of the mesh directions computed on continuous property on nearshore$_a$ environment.

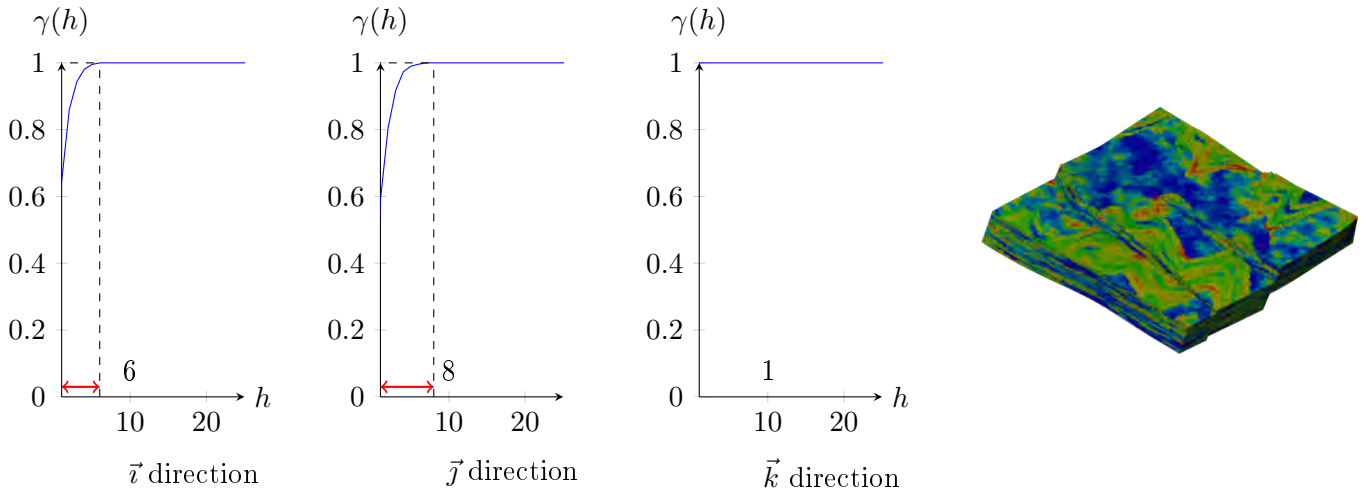


Figure 5.15: Three variograms of the mesh directions computed on continuous property on fluvial environment.

Subsection 4.2.3: Continuous properties, which precision?

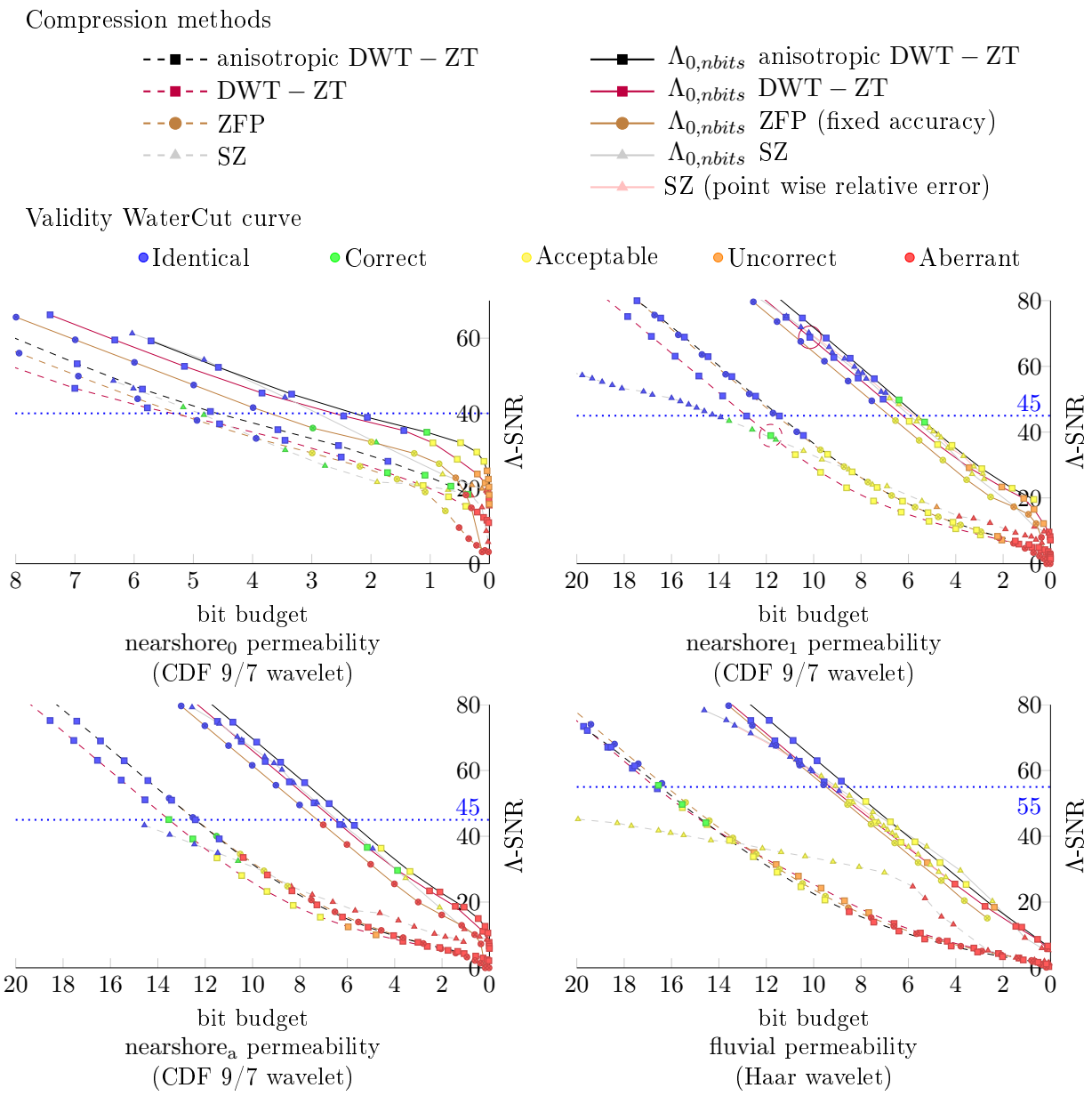


Figure 5.16: Permeability from nearshore$_0$ (top on the left) end nearshore$_1$ (top on the right) nearshore$_a$ (bottom on the left) and fluvial (bottom on the right) environments generated at refinable precision for decreasing bit budget (a mark per precision) by our alternative (wavelet packet or dyadic decomposition), SZ and ZFP (using or not Λ_0), objectively evaluated by Λ-SNR; highlighted by subjective appreciation of simulation results (color code).

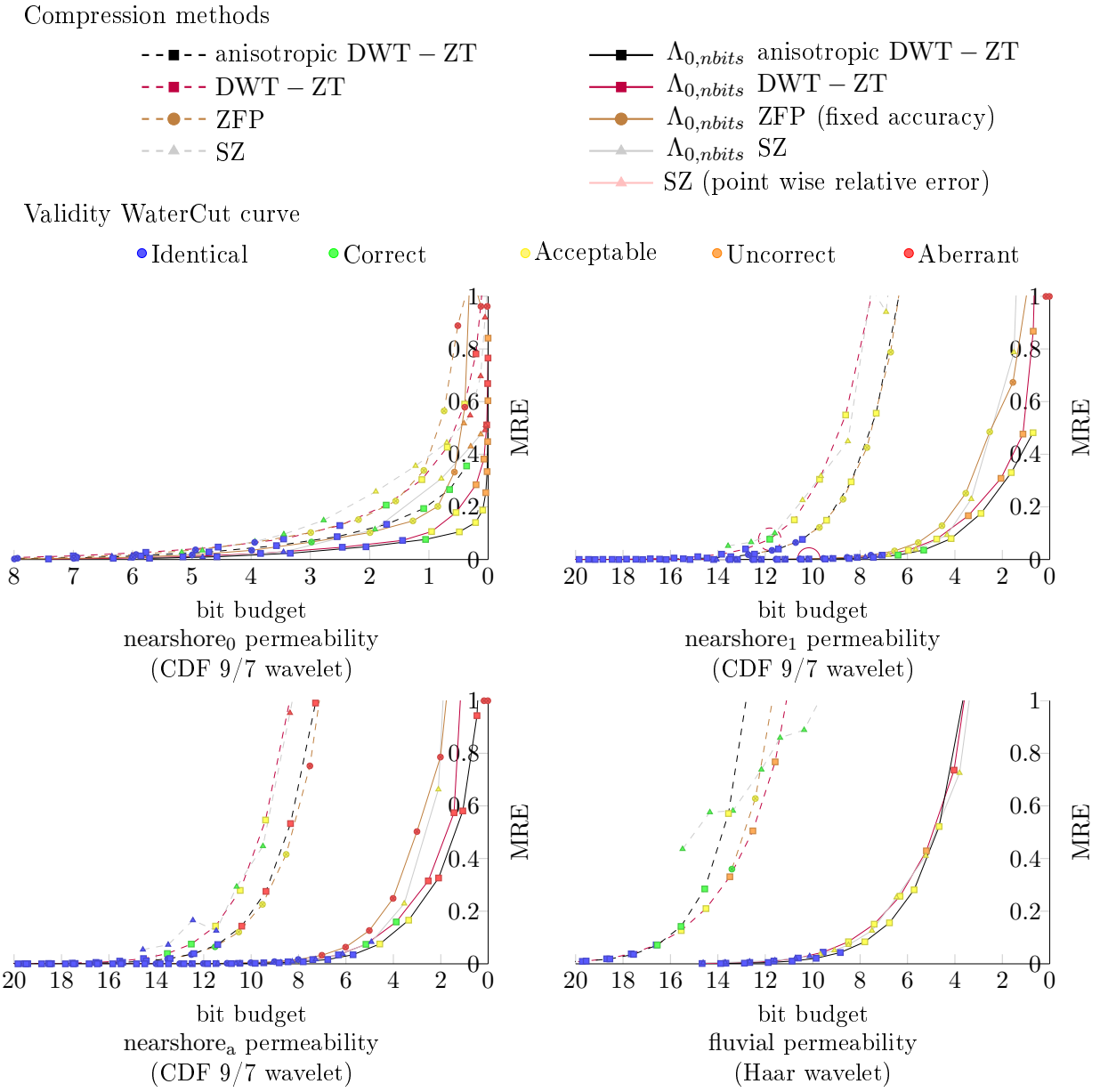


Figure 5.17: Permeability from nearshore$_0$ (top on the left) end nearshore$_1$ (top on the right) nearshore$_a$ (bottom on the left) and fluvial (bottom on the right) environments generated at refinable precision for decreasing bit budget (a mark per precision) by our alternative (wavelet packet or dyadic decomposition), SZ and ZFP (using or not Λ_0), objectively evaluated by MRE; highlighted by subjective appreciation of simulation results (color code).

Subsection 4.3.2: Thresholding at necessary precision

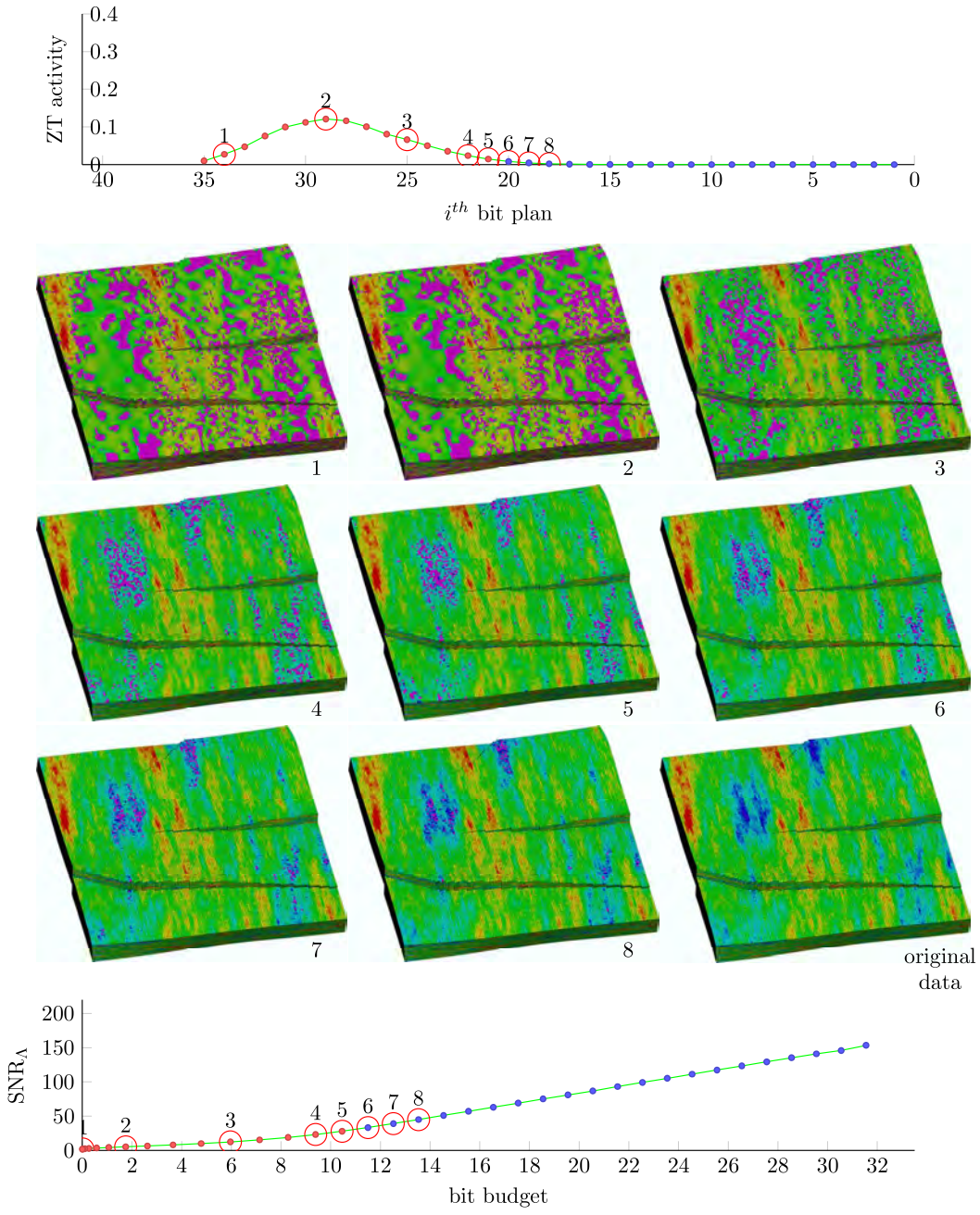


Figure 5.18: Correlation between ZT activity (top-curve) and objective evaluation by Λ-SNR (bottom-curve) of permeability from $\text{nearshore}_\text{a}$ environment generated at refinable precision by increasing bit planes number for reconstruction with our compression alternative using Λ_1.

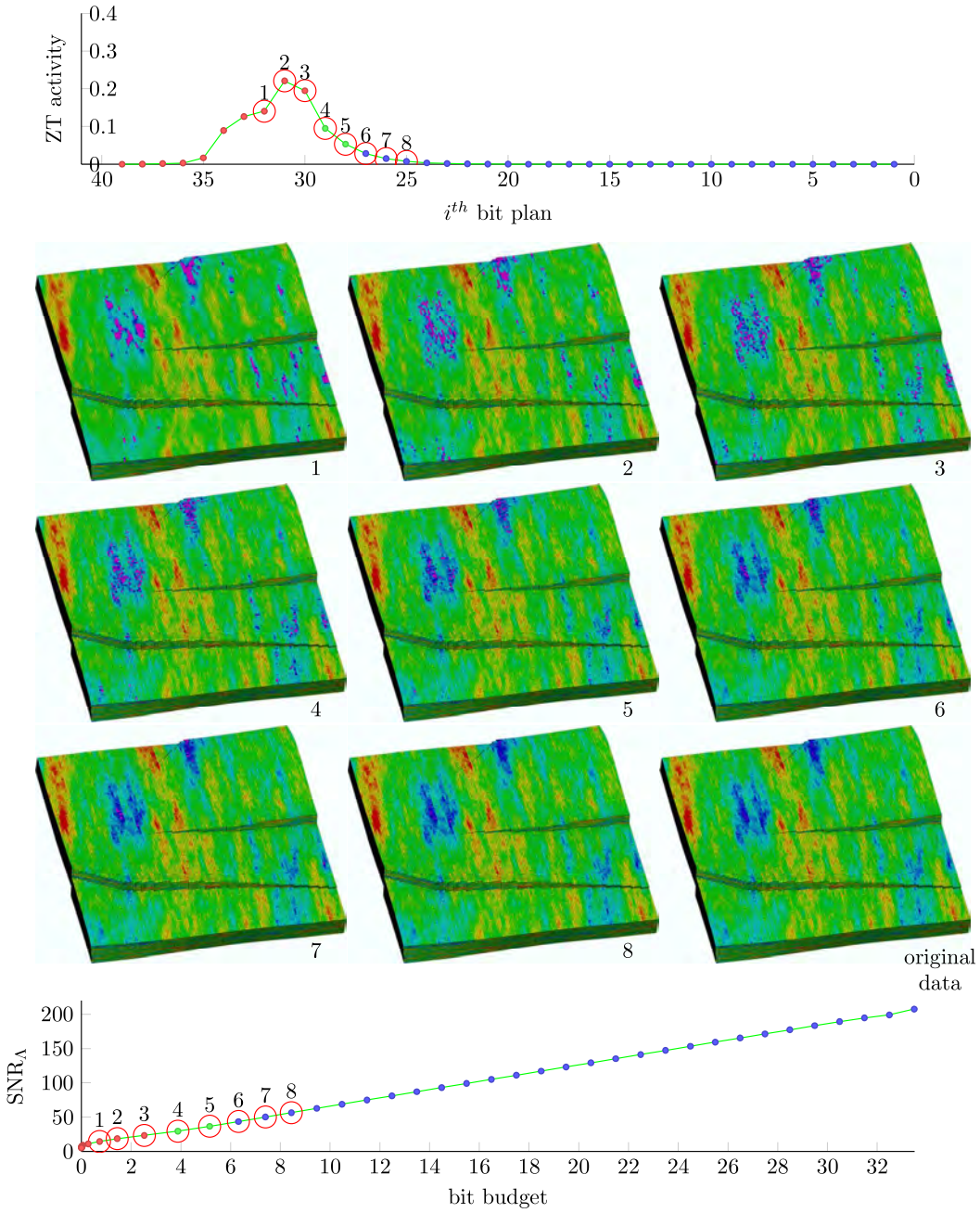


Figure 5.19: Correlation between ZT activity (top-curve) and objective evaluation by Λ-SNR (bottom-curve) of permeability from $\text{nearshore}_\text{a}$ environment generated at refinable precision by increasing bit planes number for reconstruction with our compression alternative using Λ_0.

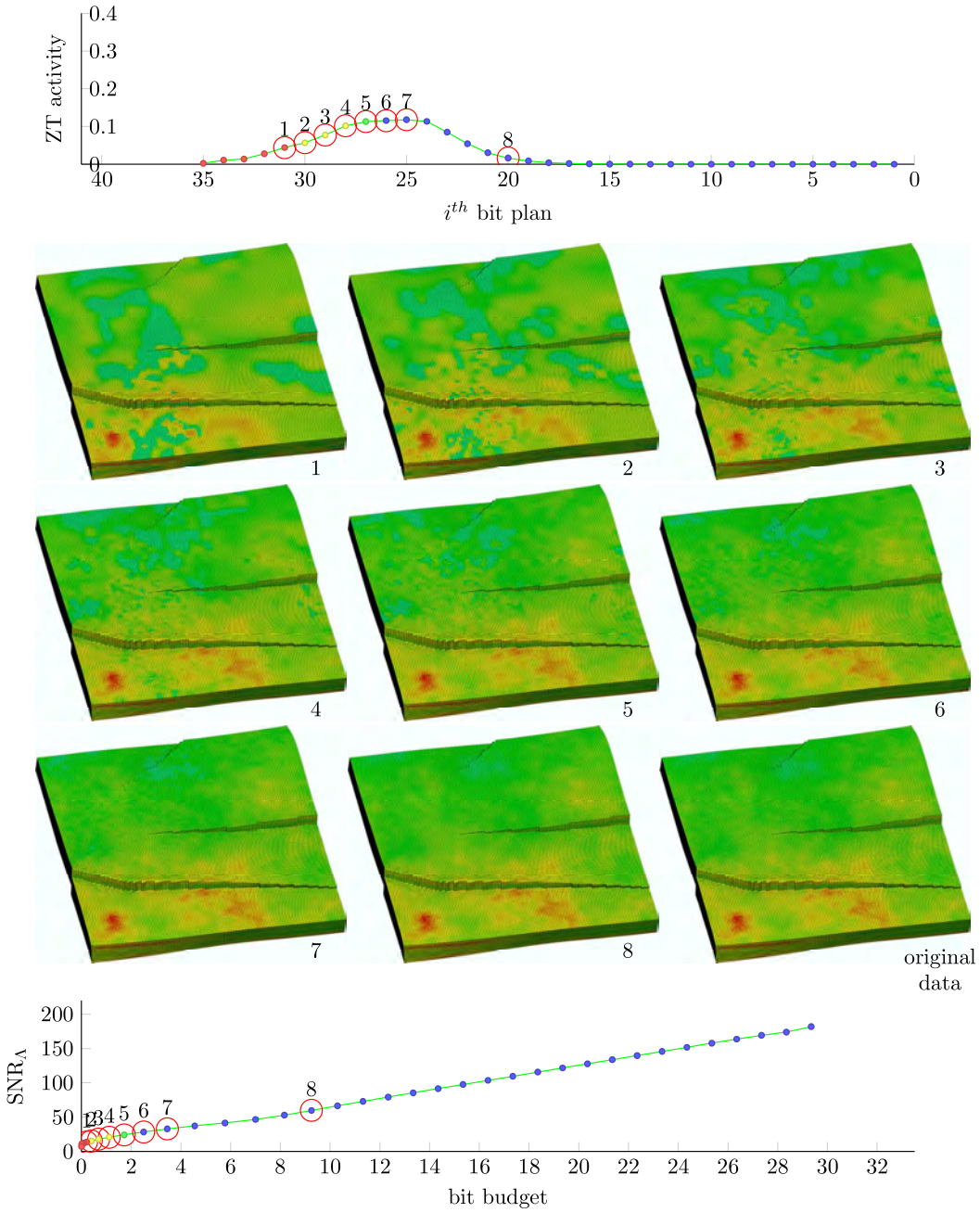


Figure 5.20: Correlation between ZT activity (top-curve) and objective evaluation by Λ-SNR (bottom-curve) of permeability from nearshore_0 environment generated at refinable precision by increasing bit planes number for reconstruction with our compression alternative using Λ_1.

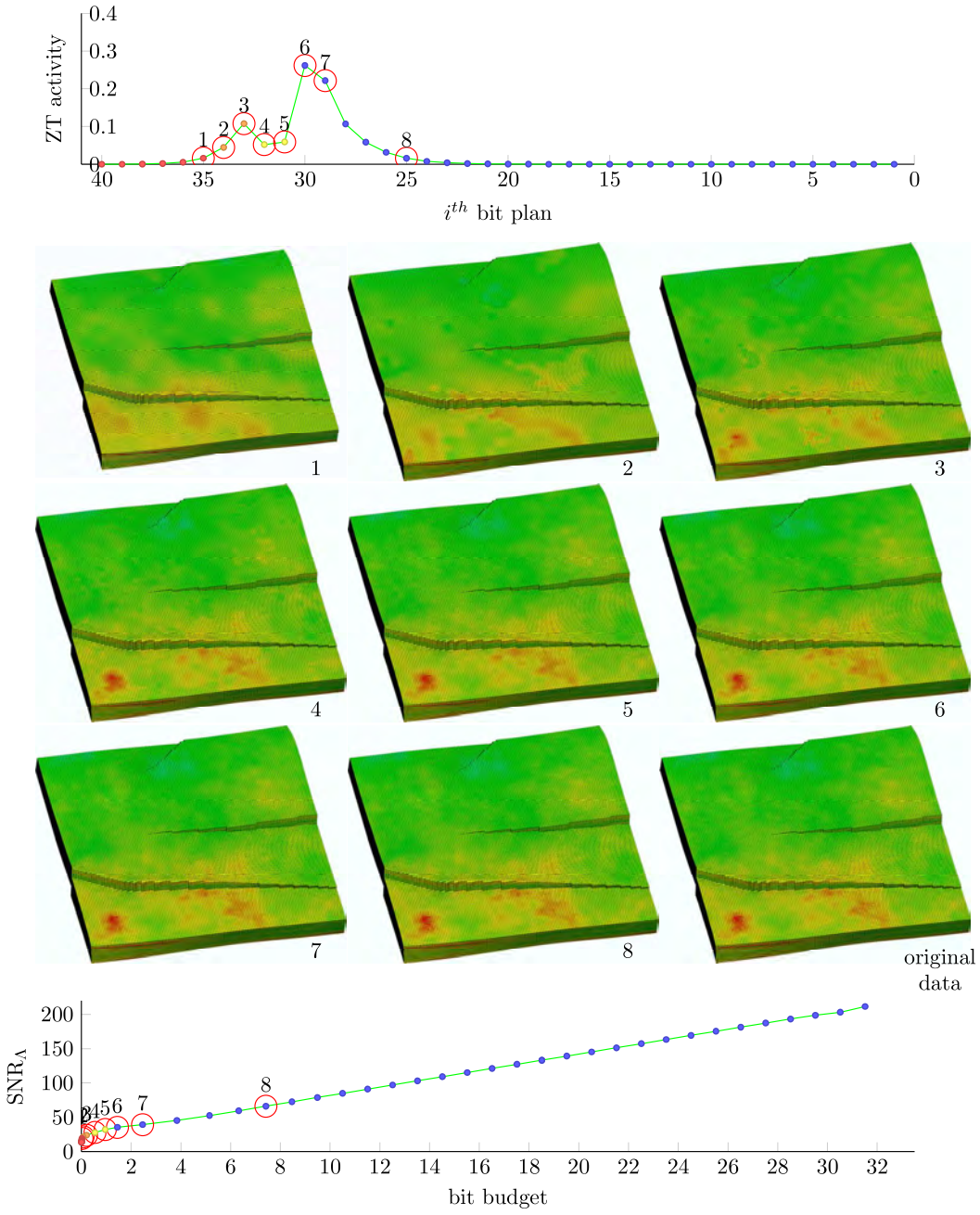


Figure 5.21: Correlation between ZT activity (top-curve) and objective evaluation by Λ-SNR (bottom-curve) of permeability from nearshore$_0$ environment generated at refinable precision by increasing bit planes number for reconstruction with our compression alternative using Λ_0.

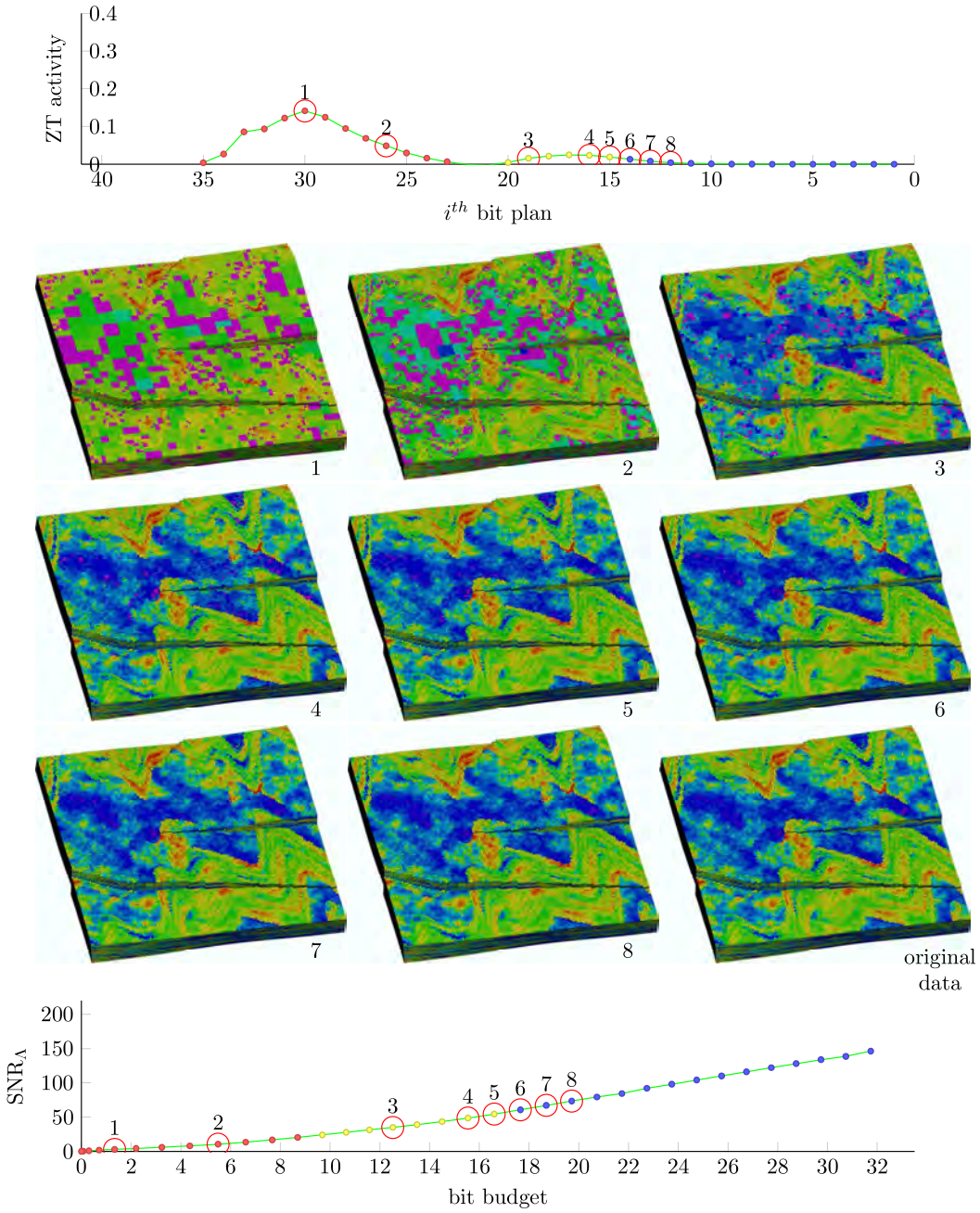


Figure 5.22: Correlation between ZT activity (top-curve) and objective evaluation by Λ-SNR (bottom-curve) of permeability from fluvial environment generated at refinable precision by increasing bit planes number for reconstruction with our compression alternative using Λ_1.

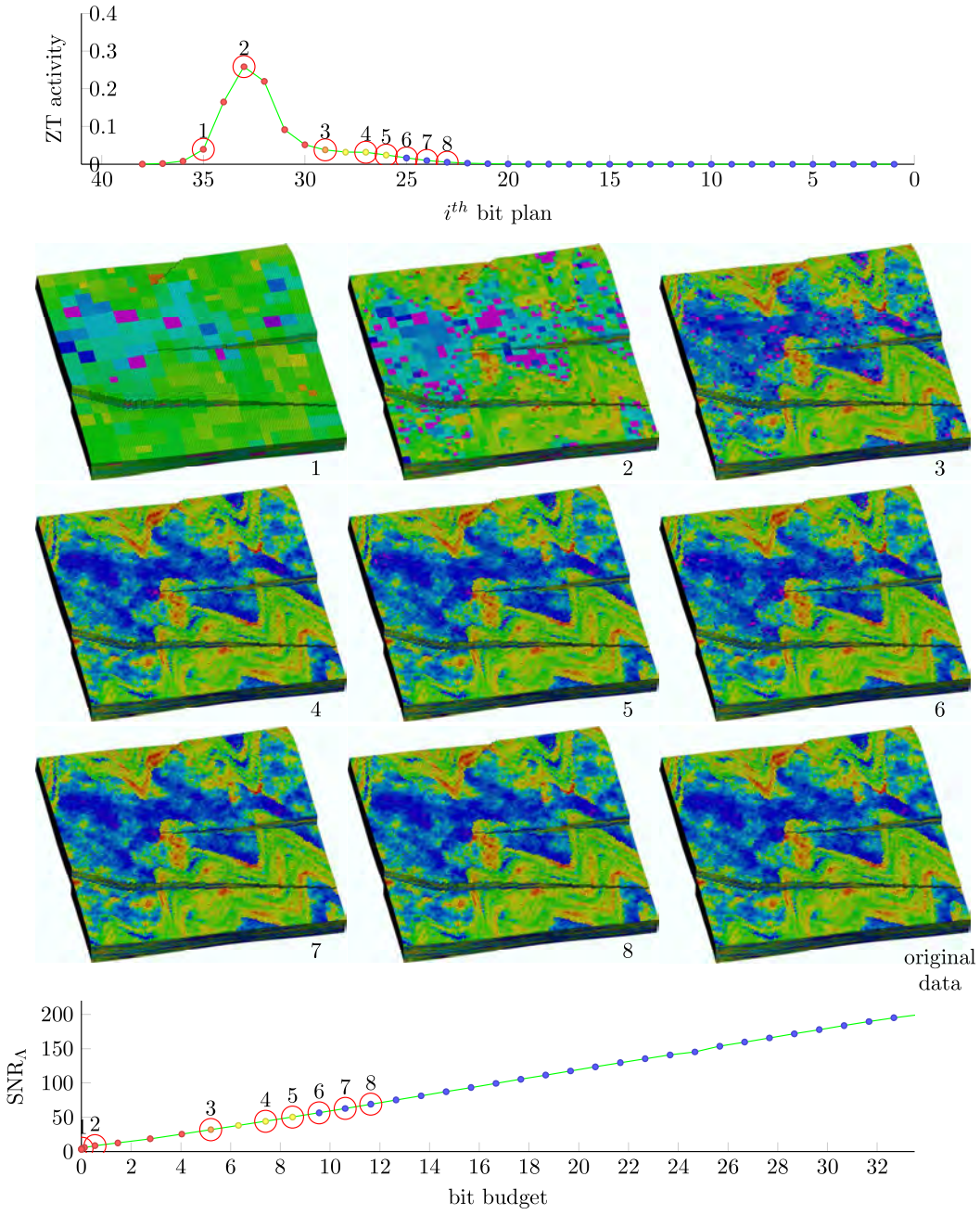


Figure 5.23: Correlation between ZT activity (top-curve) and objective evaluation by Λ-SNR (bottom-curve) of permeability from fluvial environment generated at refinable precision by increasing bit planes number for reconstruction with our compression alternative using Λ_0.

Bibliography

Ahvar, E., Orgerie, A.-C., and Lebre, A. (2019). Estimating energy consumption of cloud, fog and edge computing infrastructures. *IEEE Trans. Sustain. Comput.*, pages 1–1. 7

Ainsworth, M., Tugluk, O., Whitney, B., and Klasky, S. (2018). Multilevel techniques for compression and reduction of scientific data—the univariate case. *Computing and Visualization in Science*, 19(5-6):65–76. 38

Ainsworth, M., Tugluk, O., Whitney, B., and Klasky, S. (2019). Multilevel techniques for compression and reduction of scientific data-quantitative control of accuracy in derived quantities. *SIAM J. Sci. Comput. [Correct to j-siam-j-sci-comp]*, 41(4):A2146–A2171. 38

Ainsworth, M., Tugluk, O., Whitney, B., and Klasky, S. (2020a). Multilevel techniques for compression and reduction of scientific data—the multivariate case. *SIAM J. Sci. Comput.*, 41(2):A1278–A1303. 38

Ainsworth, M., Tugluk, O., Whitney, B., and Klasky, S. (2020b). Multilevel techniques for compression and reduction of scientific data—the unstructured case. *SIAM Journal on Scientific Computing*, 42(2):A1402–A1427. 38

Antonini, M., Payan, F., Schneider, S., Duval, L., and Peyrot, J.-L. (2017). Method of exploitation of hydrocarbons of an underground formation by means of optimized scaling. Patent. 48

Babaei, M. (2013). *Multiscale Wavelet and Upscaling-Downscaling for Reservoir Simulation.* PhD thesis, Imperial College London, Departement of Earth Science and Engineering. 31

Baker, A. H., Hammerling, D. M., Mickelson, S. A., Xu, H., Stolpe, M. B., Naveau, P., Sanderson, B., Ebert-Uphoff, I., Samarasinghe, S., Simone, F. D., Carbone, F., Gencarelli, C. N., Dennis, J. M., Kay, J. E., and Lindstrom, P. (2016). Evaluating lossy data compression on climate simulation data within a large ensemble. *Geosci. Model Dev.*, 9(12):4381–4403. 78

Baker, A. H., Hammerling, D. M., and Turton, T. L. (2019). Evaluating image quality measures to assess the impact of lossy data compression applied to climate simulation data. *Comput. Graph. Forum*, 38(3):517–528. 36, 78, 82

Baker, A. H., Xu, H., Dennis, J. M., Levy, M. N., Nychka, D., Mickelson, S. A., Edwards, J., Vertenstein, M., and Wegener, A. (2014). A methodology for evaluating the impact of data compression on climate simulation data. In *Proceedings of the 23rd international symposium on High-performance parallel and distributed computing - HPDC 14*. ACM Press. 37, 74, 77, 78

Baker, A. H., Xu, H., Hammerling, D. M., Li, S., and Clyne, J. P. (2017). Toward a multi-method approach: Lossy data compression for climate simulation data. In *Lecture Notes in Computer Science*, pages 30–42. Springer International Publishing. 36, 77, 82

Ballester-Ripoll, R., Lindstrom, P., and Pajarola, R. (2019). TTHRESH: Tensor compression for multidimensional visual data. *IEEE Trans. Visual Comput. Graph.*, pages 1–1. ix, 78, 79, 81

Bey, J. (1995). Tetrahedral grid refinement. *Computing*, 55(4):355–378. 34

Boscardín, L. B., Castro, L. R., Castro, S. M., and Giusti, A. D. (2006). Wavelets bases defined over tetrahedra. *INSTEC J. Computer Science and Technology*, 6(1):46–52. 34

Bouard, L. (2017). Generation of DEM packings from RingMesh geological models. 77

Bouard, L., Duval, L., Payan, F., and Antonini, M. (2018). Décomposition multi-échelles de maillages 3D hexaédriques dans le domaine des géosciences. Étude des performances en compression sans pertes. In *Proc. colloque COmpression et REprésentation des Signaux Audiovisuels*. 35

Bradley, J. and Brislawn, C. (1993). Wavelet transform-vector quantization compression of supercomputer ocean models. In *Proc. Data Compression Conf.* IEEE Comput. Soc. Press. 36, 37

Bruekers, F. A. M. L. and van den Enden, A. W. M. (1992). New networks for perfect inversion and perfect reconstruction. *IEEE J. Sel. Areas Comm.*, 10(1):129–137. 40

Burrows, M. and Wheeler, D. J. (1994). A block-sorting lossless data compression algorithm. Technical report, Digital Systems Research Center. 55

Calderbank, A. R., Daubechies, I., Sweldens, W., and Yeo, B.-L. (1998). Wavelet transforms that map integers to integers. *Appl. Comput. Harmon. Analysis*, 5(3):332–369. 46

Calhoun, J., Cappello, F., Olson, L. N., Snir, M., and Gropp, W. D. (2018). Exploring the feasibility of lossy compression for PDE simulations. page 109434201876203. 79, 80

Cappello, F., Di, S., Li, S., Liang, X., Gok, A. M., Tao, D., Yoon, C. H., Wu, X.-C., Alexeev, Y., and Chong, F. T. (2019). Use cases of lossy compression for floating-point data in scientific data sets. *Int. J. High Perform. Comput. Appl.* ix, 7, 38, 62, 69, 78, 79, 80

Chaux, C., Duval, L., Benazza-Benyahia, A., and Pesquet, J.-C. (2008). A nonlinear Stein based estimator for multichannel image denoising. *IEEE Trans. Signal Process.*, 56(8):3855–3870. 40

Chaux, C., Pesquet, J.-C., and Duval, L. (2007). Noise covariance properties in dual-tree wavelet decompositions. *IEEE Trans. Inform. Theory*, 53(12):4680–4700. 40

Chen, D., Chiang, Y.-J., Memon, N., and Wu, X. (2005). Geometry compression of tetrahedral meshes using optimized prediction. In *Proc. Eur. Sig. Image Proc. Conf.* 33

Chen, Z., Son, S. W., Hendrix, W., Agrawal, A., Liao, W.-K., and Choudhary, A. (2014). NUMARCK: Machine learning algorithm for resiliency and checkpointing. In *Proc. Int. Conf. High Perf. Comput., Netw., Storage Anal.* IEEE. 78

Chizat, L. (2014). Multiresolution signal compression: Exploration and application. Master's thesis, ENS Cachan. 35, 113

Christie, M. and Blunt, M. (2001). Tenth SPE comparative solution project: A comparison of upscaling techniques. In *SPE Reservoir Simulation Symposium*. Society of Petroleum Engineers. viii, 64

Christophe, E. (2006). *Compression des Images Hyperspectrales et son Impact sur la Qualité des Données*. PhD thesis. 102

Christophe, E., Mailhes, C., and Duhamel, P. (2008). Hyperspectral image compression: Adapting SPIHT and EZW to anisotropic 3-D wavelet coding. *IEEE Trans. Image Process.*, 17(12):2334–2346. ix, 62, 99

Clyne, J., Mininni, P., Norton, A., and Rast, M. (2007). Interactive desktop analysis of high resolution simulations: Application to turbulent plume dynamics and current sheet formation. *New Journal of Physics*, 9(8):301–301. 37

Coats, K., Nielsen, R., Terhune, M. H., and Weber, A. (1967). Simulation of three-dimensional, two-phase flow in oil and gas reservoirs. *Society of Petroleum Engineers Journal*, 7(04):377–388. 30

Cohen, A., Daubechies, I., and Feauveau, J.-C. (1992). Biorthogonal bases of compactly supported wavelets. *Commun. ACM*, 45(5):485–560. 46, 90

Courbet, C. and Isenburg, M. (2010). Streaming compression of hexahedral meshes. *Vis. Comput.*, 26(6-8):1113–1122. 33

Cunningham, J. P. and Ghahramani, Z. (2014). Unifying linear dimensionality reduction. *PREPRINT*. 35

Danovaro, E., De Floriani, L., Lee, M. T., and Samet, H. (2002). Multiresolution tetrahedral meshes: An analysis and a comparison. In *Proc. Shape Modeling International*, pages 83–91. 34

Delaunay, X., Courtois, A., and Gouillon, F. (2019). Evaluation of lossless and lossy algorithms for the compression of scientific datasets in netCDF-4 or HDF5 files. *Geosci. Model Dev.*, 12(9):4099–4113. 36, 77, 78

Dennis, J. M., Edwards, J., Evans, K. J., Guba, O., Lauritzen, P. H., Mirin, A. A., St-Cyr, A., Taylor, M. A., and Worley, P. H. (2011). CAM-SE: A scalable spectral element dynamical core for the community atmosphere model. *Int. J. High Perform. Comput. Appl.*, 26(1):74–89. 78

Deutsch, L. (1996). DEFLATE compressed data format specification version 1.3. Technical report. 33

Di, S. and Cappello, F. (2016). Fast error-bounded lossy HPC data compression with SZ. In *IEEE International Parallel and Distributed Processing Symposium*. IEEE. 80

Diffenderfer, J., Fox, A. L., Hittinger, J. A., Sanders, G., and Lindstrom, P. G. (2019). Error analysis of ZFP compression for floating-point data. *SIAM J. Sci. Comput. [Correct to j-siam-j-sci-comp]*, 41(3):A1867–A1898. 38, 80

Durlofsky, L., Milliken, W., and Bernath, A. (2000). Scaleup in the near-well region. *SPE Journal*, 5(01):110–117. 30

Durlofsky, L. J. and Chen, Y. (2012). *Uncertainty quantification for subsurface flow problems using coarse-scale models*. Springer. 30

Duwe, K., Lüttgau, J., Mania, G., Squar, J., Fuchs, A., Kuhn, M., Betke, E., and Ludwig, T. (2020). State of the art and future trends in data reduction for high-performance computing. *Supercomput. Front. Innov.*, 10(1). 35

Eyring, V., Bony, S., Meehl, G. A., Senior, C. A., Stevens, B., Stouffer, R. J., and Taylor, K. E. (2016). Overview of the coupled model intercomparison project phase 6 (CMIP6) experimental design and organization. *Geoscientific Model Development*, 9(5):1937–1958. 76, 78

Fischer, P., Lottes, J., and Kerkemeier, S. (2008). Nek5000: Open source spectral element cfd solver. 79

Fowler, A. C. and Yang, X.-S. (2003). Dissolution/precipitation mechanisms for diagenesis in sedimentary basins. *Journal of Geophysical Research: Solid Earth*, 108(B10). 98

Fowler, J. E. (2000). QccPack: an open-source software library for quantization, compression, and coding. In *Proc. Data Compression Conf.*, page 554. 37, 62

Fox, A., Diffenderfer, J., Hittinger, J., Sanders, G., and Lindstrom, P. (2020). Stability analysis of inline zfp compression for floating-point data in iterative methods. 80

García-Martín, E., Rodrigues, C. F., Riley, G., and Grahn, H. (2019). Estimation of energy consumption in machine learning. *J. Parallel Distrib. Comput.*, 134:75–88. 7

Geist, A. and Reed, D. A. (2016). A survey of high-performance computing scaling challenges. *Int. J. High Perform. Comput. Appl.*, 31(1):104–113. 8, 35

Geveler, M., Köhler, M., Ribbrock, D., Saak, J., Truschkewitz, G., Benner, P., and Turek, S. (2015). Future data centers for energy-efficient large scale numerical simulations. In *7th KoMSO Challenge Workshop*. 7

Gok, A. M., Di, S., Alexeev, Y., Tao, D., Mironov, V., Liang, X., and Cappello, F. (2018). PaSTRI: Error-bounded lossy compression for two-electron integrals in quantum chemistry. IEEE. 80

Grout, R., Gruber, A., Yoo, C., and Chen, J. (2011). Direct numerical simulation of flame stabilization downstream of a transverse fuel jet in cross-flow. *Proc. Combust. Inst.*, 33(1):1629–1637. 79

Gumhold, S., Guthe, S., and Straßer, W. (1999). Tetrahedral mesh compression with the cut-border machine. In *Proc. IEEE Visualization Conf.*, pages 51–58. 32

Gumhold, S. and Straßer, W. (1998). Real time compression of triangle mesh connectivity. In *Proc. ACM SIGGRAPH Comput. Graph.*, pages 133–140. 32

Haar, A. (1910). Zur Theorie der orthogalen Funktionen Systeme. *Math. Annalen*, 69:331–371. 59

Hadjidoukas, P. and Wermelinger, F. (2019). A parallel data compression framework for large scale 3d scientific data. 39, 79

Hoang, D., Summa, B., Bhatia, H., Lindstrom, P., Klacansky, P., Usher, W., Bremer, P.-T., and Pascucci, V. (2021). Efficient and flexible hierarchical data layouts for a unified encoding of scalar field precision and resolution. *IEEE Trans. Visual Comput. Graph.* 97

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., and Stuetzle, W. (1993). Mesh optimization. In *Proc. ACM SIGGRAPH Comput. Graph.*, pages 19–26. 34

Hübbe, N. and Kunkel, J. (2012). Reducing the HPC-datastorage footprint with MAFISC—multidimensional adaptive filtering improved scientific data compression. *Computer Science - Research and Development*, 28(2-3):231–239. 36

Hübbe, N., Wegener, A., Kunkel, J. M., Ling, Y., and Ludwig, T. (2013). Evaluating lossy compression on climate data. In *Lecture Notes in Computer Science*, pages 343–356. Springer Berlin Heidelberg. 7, 37

Huffman, D. A. (1952). A method for the construction of minimum-redundancy codes. *Proc. IRE*, 40(9):1098–1101. 10

Ibarria, L., Lindstrom, P., and Rossignac, J. (2007). Spectral predictors. In *Proc. Data Compression Conf.*, pages 163–172. 33

Isenburg, M. and Alliez, P. (2003). Compressing hexahedral volume meshes. *Graph. Model.*, 65(4):239–257. 32, 33

Isenburg, M., Lindstrom, P., Gumhold, S., and Shewchuk, J. (2006). Streaming compression of tetrahedral volume meshes. In *Proc. Graphics Interface*, pages 115–121. 33

Iverson, J., Kamath, C., and Karypis, G. (2012). Fast and effective lossy compression algorithms for scientific datasets. In *ProEuro-Par Parallel Processing*, pages 843–856. Springer Berlin Heidelberg. 31, 38, 39

Jacques, L., Duval, L., Chaux, C., and Peyré, G. (2011). A panorama on multiscale geometric representations, intertwining spatial, directional and frequency selectivity. *Signal Process.*, 91(12):2699–2730. 34, 40, 59

Jensen, J., Lake, L. W., Corbett, P. W., and Goggin, D. (2000). *Statistics for petroleum engineers and geoscientists*. Gulf Professional Publishing. 30

Karlin, I. (2012). LULESH programming model and performance ports overview. Technical report. 79

Kipnis, A., Eldar, Y. C., and Goldsmith, A. J. (2018). Analog-to-digital compression: A new paradigm for converting signals to bits. *IEEE Signal Process. Mag.*, 35(3):16–39. 36

Kolomenskiy, D., Onishi, R., and Uehara, H. (2018). Data compression for environmental flow simulations. *PREPRINT*. Open source code. 8, 37

Koranne, S. (2011). *Handbook of Open Source Tools*. Spring. 36

Kovačević, J., Goyal, V., and Vetterli, M. (2012). *Signal Processing Fourier and Wavelet Representations*. 40

Kraska, T., Beutel, A., Chi, E. H., Dean, J., and Polyzotis, N. (2018). The case for learned index structures. In *Proceedings of the 2018 International Conference on Management of Data - SIGMOD '18*. ACM Press. 8, 35

Krivograd, S., Trlep, M., and Žalik, B. (2008). A hexahedral mesh connectivity compression with vertex degrees. *Comput. Aided Des.*, 40(12):1105–1112. 33

Kuhn, M., Kunkel, J. M., and Ludwig, T. (2016). Data compression for climate data. *Supercomputing Frontiers and Innovations*, 3(1). 7

Kunkel, J., Kuhn, M., and Ludwig, T. (2014). Exascale storage systems an analytical study of expenses. *Supercomputing Frontiers and Innovations*, 1(1). 7

Kunkel, J., Novikova, A., Betke, E., and Schaare, A. (2017). Toward decoupling the selection of compression algorithms from quality constraints. In *Lect. Notes Comput. Sci.*, pages 3–14. Springer International Publishing. 39

Lakshminarasimhan, S., Shah, N., Ethier, S., Klasky, S., Latham, R., Ross, R., and Samatova, N. F. (2011). Compressing the incompressible with ISABELA: In-situ reduction of spatio-temporal data. In *Euro-Par 2011 Parallel Processing*, pages 366–379. Springer Berlin Heidelberg. 38

Laney, D., Langer, S., Weber, C., Lindstrom, P., and Wegener, A. (2014). Assessing the effects of data compression in simulations using physically motivated metrics. *Sci. Program.*, 22:141–155. 38, 79, 81, 82

Le Gall, D. and Tabatabai, A. (1988). Sub-band coding of digital images using symmetric short kernel filters and arithmetic coding techniques. In *Proc. Int. Conf. Acoust. Speech Signal Process.* 46

Li, D. and Beckner, B. (2000). Optimal uplayering for scaleup of multimillion-cell geologic models. In *SPE Annual Technical Conference and Exhibition*. Society of Petroleum Engineers. 30

Li, S., Gruchalla, K., Potter, K., Clyne, J., and Childs, H. (2015). Evaluating the efficacy of wavelet configurations on turbulent-flow data. In *Proc. IEEE Symp. Large Data Anal. Visual.*, pages 81–89, Chicago, IL, USA. 37, 82

Li, S., Jaroszynski, S., Pearse, S., Orf, L., and Clyne, J. (2019). Vapor: A visualization package tailored to analyze simulation data in earth system science. *Atmosphere 2019, 10, 488*. 37

Li, S., Larsen, M., Clyne, J., and Childs, H. (2017a). Performance impacts of in situ wavelet compression on scientific simulations. In *Proceedings of the In Situ Infrastructures on Enabling Extreme-Scale Analysis and Visualization - ISAV '17*. ACM Press. 79

Li, S., Marsaglia, N., Garth, C., Woodring, J., Clyne, J., and Childs, H. (2018). Data reduction techniques for simulation, visualization and data analysis. *Comput. Graph. Forum*, 37(6):422–447. 35

Li, S., Sane, S., Orf, L., Mininni, P., Clyne, J., and Childs, H. (2017b). Spatiotemporal wavelet compression for visualization of scientific simulation data. In *2017 IEEE International Conference on Cluster Computing (CLUSTER)*. IEEE. 37

Li, Y., Perlman, E., Wan, M., Yang, Y., Meneveau, C., Burns, R., Chen, S., Szalay, A., and Eyink, G. (2008). A public turbulence database cluster and applications to study lagrangian evolution of velocity increments in turbulence. *Journal of Turbulence*, 9:N31. 78

Li, Y., Sharan, L., and Adelson, E. H. (2005). Compressing and companding high dynamic range images with subband architectures. *ACM Trans. Graph.*, 24(3):836. 23

Liang, X., Di, S., Tao, D., Chen, Z., and Cappello, F. (2018a). An efficient transformation scheme for lossy data compression with point-wise relative error bound. In *2018 IEEE International Conference on Cluster Computing (CLUSTER)*. IEEE. 38

Liang, X., Di, S., Tao, D., Li, S., Li, S., Guo, H., Chen, Z., and Cappello, F. (2018b). Error-controlled lossy compression optimized for high compression ratios of scientific datasets. In *IEEE Int. Conf. Big Data*. 80

Lie, K.-A. (2016). *An Introduction to Reservoir Simulation Using MATLAB. User Guide for the Matlab Reservoir Simulation Toolbox (MRST)*. SINTEF ICT, Departement of Applied Mathematics. 21, 84

Lindstrom, P. (2014). Fixed-rate compressed floating-point arrays. *IEEE Trans. Visual Comput. Graph.*, 20(12):2674–2683. ix, 38, 80

Lindstrom, P., Chen, P., and Lee, E.-J. (2016). Reducing disk storage of full-3D seismic waveform tomography (f3dt) through lossy online compression. *Comput. Geosci.* 39, 80

Lindstrom, P. and Isenburg, M. (2006). Fast and efficient compression of floating-point data. *IEEE Trans. Visual Comput. Graph.*, 12(5):1245–1250. 37

Lindstrom, P. and Isenburg, M. (2008). Lossless compression of hexahedral meshes. In *Proc. Data Compression Conf.*, pages 192–201. 33

Lindstrom, P., Lloyd, S., and Hittinger, J. (2018). Universal coding of the reals: Alternatives to IEEE floating point. In *Proc. Conference for Next Generation Arithmetic (CONGA)*. ACM Press. 62

Loddoch, A. and Schmalzl, J. (2006). Variable quality compression of fluid dynamical data sets using a 3-d DCT technique. *Geochemistry, Geophysics, Geosystems*, 7(1):n/a–n/a. 37

Meehl, G. A., Covey, C., and McAvaney, B. (2005). Overview of the coupled model intercomparison project. In *Bulletin of the American Meteorological Society*, volume 86, pages 89–93. 76

Mehrabi, A. R. and Sahimi, M. (1997). Coarsening of heterogeneous media: Application of wavelets. *Physical Review Letters*, 79(22):4385–4388. 31

Muller, J.-M., Brunie, N., de Dinechin, F., Jeannerod, C.-P., Joldes, M., Lefèvre, V., Melquiond, G., Revol, N., and Torres, S. (2018). *Handbook of Floating-Point Arithmetic*. Springer. 62

Nakahashi, K. (2005). High-density mesh flow computations with pre-/post-data compressions. In *17th AIAA Computational Fluid Dynamics Conference*. American Institute of Aeronautics and Astronautics. 37

Naksinehaboon, N., Liu, Y., Leangsuksun, C., Nassar, R., Paun, M., and Scott, S. (2008). Reliability-aware approach: An incremental checkpoint/restart model in HPC environments. In *2008 Eighth IEEE International Symposium on Cluster Computing and the Grid (CCGRID)*. IEEE. 4

Natarajan, B. K. (1993). Filtering random noise via data compression. In *Proc. Data Compression Conf.*, pages 60–69. 105

Ohm, J.-R. and Sullivan, G. J. (2013). High efficiency video coding: The next frontier in video compression. *IEEE Signal Process. Mag.*, 30(1):152–158. 10

Otero, E., Vinuesa, R., Schlatter, P., Marin, O., Siegel, A., and Laure, E. (2019). The effect of lossy data compression in computational fluid dynamics applications: Resilience and data postprocessing. In *Direct and Large-Eddy Simulation XI*, pages 175–181. Springer International Publishing. 79, 81

Pajarola, R., Rossignac, J., and Szymczak, A. (1999). Implant sprays: Compression of progressive tetrahedral mesh connectivity. In *Proc. IEEE Visualization Conf.*, pages 299–305. 34

Pavlichin, D., Weissman, T., and Mably, G. (2018). The quest to save genomics: Unless researchers solve the looming data compression problem, biomedical science could stagnate. *IEEE Spectrum*, 55(9):27–31. 6

Peaceman, D. W. (1977). *Fundamentals of numerical reservoir simulation*. Elsevier. 30

Peyrot, J.-L., Duval, L., Payan, F., Bouard, L., Chizat, L., Schneider, S., and Antonini, M. (2019). HexaShrink, an exact scalable framework for hexahedral meshes with attributes and discontinuities: multiresolution rendering and storage of geoscience models. *Computat. Geosci.*, 23:723–743. v, 35, 40, 69

Peyrot, J.-L., Duval, L., Schneider, S., Payan, F., and Antonini, M. (2016). (H)exashrink: Multiresolution compression of large structured hexahedral meshes with discontinuities in geosciences. In *Proc. Int. Conf. Image Process.*, pages 1101–1105. 40

Pigeon, S. and Bengio, Y. (1999). Binary pseudowavelets and applications to bilevel image processing. In *Proc. Data Compression Conf.*, pages 364 –373. 58

Ponting, D. K. (1989). Corner point geometry in reservoir simulation. In *ECMOR I - 1st European Conference on the Mathematics of Oil Recovery*. EAGE Publications BV. 21

Preux, C. (2014). About the use of quality indicators to reduce information loss when performing upscaling. *Oil Gas Sci. Tech.*, 71(1):7. 31, 86

Pulido, J., Livescu, D., Woodring, J., Ahrens, J., and Hamann, B. (2016). Survey and analysis of multiresolution methods for turbulence data. *Comput. Fluids*, 125:39–58. 82

Qi, D. and Zhang, S. (2009). Major challenges for reservoir upscaling. *Petroleum Science and Technology*, 27(17):1985–1992. 30

Raïs, I., Balouek-Thomert, D., Orgerie, A.-C., and Parashar, L. L. M. (2019). Leveraging energy-efficient non-lossy compression for data-intensive applications. *Proc. 17th International Conference on High Performance Computing & Simulation*. 7

Rao, R. M. and Bopardikar, A. S. (1998). *Wavelet Transforms: Introduction to Theory and Applications*. Prentice Hall. 40

Rasaei, M. and Sahimi, M. (2008). Upscaling and simulation of waterflooding in heterogeneous reservoirs using wavelet transformations: Application to the SPE-10 model. *Transp. Porous Med.*, 72(3):311–338. 31

Ratanaworabhan, P., Ke, J., and Burtscher, M. (2006). Fast lossless compression of scientific floating-point data. In *Proc. Data Compression Conf.*, pages 133–142. 36

Rezapour, A., Ortega, A., and Sahimi, M. (2019). Upscaling of geological models of oil reservoirs with unstructured grids using lifting-based graph wavelet transforms. *Transp. Porous Med.*, 127:661–684. 31, 42

Røe, P. and Hauge, R. (2016). A volume-conserving representation of cell faces in corner point grids. *Computat. Geosci.*, 20(3):453–460. 21

Rossignac, J. (1999). Edgebreaker: Connectivity compression for triangle meshes. *IEEE Trans. Visual Comput. Graph.*, 5(1):47–61. 32

Rossinelli, D., Koumoutsakos, P., Hejazialhosseini, B., Hadjidoukas, P., Bekas, C., Curioni, A., Bertsch, A., Futral, S., Schmidt, S. J., and Adams, N. A. (2013). 11 PFLOP/s simulations of cloud cavitation collapse. In *Proc. Int. Conf. High Perf. Comput., Netw., Storage Anal.* ACM Press. 79

Said, A. and Pearlman, W. A. (1993). Reversible image compression via multiresolution representation and predictive coding. In Haskell, B. G. and Hang, H.-M., editors, *Proc. SPIE Visual Communication Image Processing*. SPIE. 59

Said, A. and Pearlman, W. A. (1996). A new, fast, and efficient image codec based on set partitioning in hierarchical trees. *IEEE Trans. Circuits Syst. Video Technol.*, 6(3):243–250. 61

Sakai, R., Sasaki, D., Obayashi, S., and Nakahashi, K. (2013). Wavelet-based data compression for flow simulation on block-structured Cartesian mesh. *Int. J. Numer. Meth. Fluids*, 73(5):462–476. 8

Salloum, M., Johnson, K. L., Bishop, J. E., Aytac, J. M., Dagel, D., and van Bloemen Waanders, B. G. (2018). Adaptive wavelet compression of large additive manufacturing experimental and simulation datasets. *Computational Mechanics*, 63(3):491–510. 37

Salomon, D. and Motta, G. (2009). *Handbook of Data Compression*. Springer. 32

Sasaki, N., Sato, K., Endo, T., and Matsuoka, S. (2015). Exploration of lossy compression for application-level checkpoint/restart. In *IEEE International Parallel and Distributed Processing Symposium*. IEEE. 37, 79, 81

Schelkens, P., Munteanu, A., Tzannes, A., and Brislawn, C. (2006). JPEG2000 part 10 — volumetric data encoding. 62

Schendel, E. R., Jin, Y., Neil, S., Chen, J., Chang, C., Ku, S.-H., Ethier, S., Klasky, S., Latham, R., Ross, R., and Samatova, N. F. (2012). ISOBAR preconditioner for effective and high-throughput lossless data compression. In *2012 IEEE 28th International Conference on Data Engineering*. IEEE. 36

Schmalzl, J. (2003). Using standard image compression algorithms to store data from computational fluid dynamics. *Comput. Geosci.*, 29(8):1021–1031. 37, 77

Shannon, C. E. (1948). A mathematical theory of communication. *Bell Syst. Tech. J.*, 27:379–423 and 623–656. 100

Shapiro, J. M. (1993). Embedded image coding using zerotrees of wavelet coefficients. *IEEE Trans. Signal Process.*, 41:3445–3462. 61, 104

Silver, J. and Zender, C. S. (2016). Layer packing tests. 78

Silver, J. D. and Zender, C. S. (2017). The compression-error trade-off for large gridded data sets. *Geosci. Model Dev.*, 10(1):413–423. 36

Smith, R. D. and Gent, P. (2002). Reference manual for the parallel ocean program (pop). Technical report, Los Alamos National Laboratory. 6, 78

Son, S. W., Chen, Z., Hendrix, W., Agrawal, A., Liao, W.-K., and Choudhary, A. (2014). Data compression for the exascale computing era — survey. *Supercomputing Frontiers and Innovations*, 1(2). 74

Staadt, O. G. and Gross, M. H. (1998). Progressive tetrahedralizations. In *Proc. IEEE Visualization Conf.*, pages 397–402. 34

Sweldens, W. (1996). The lifting scheme: a custom-design construction of biorthogonal wavelets. *Appl. Comput. Harmon. Analysis*, 3(2):186–200. 40

Szorc, G. (2017). Better compression with zstandard. 11

Szymczak, A. and Rossignac, J. (2000). Grow & fold: Compressing the connectivity of tetrahedral meshes. *Comput. Aided Des.*, 32(8-9):527–537. 32

Tao, D., Di, S., Chen, Z., and Cappello, F. (2017). Significantly improving lossy compression for scientific data sets based on multidimensional prediction and error-controlled quantization. In *2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE. 80

Tao, D., Di, S., Liang, X., Chen, Z., and Cappello, F. (2019). Optimizing lossy compression rate-distortion from automatic online selection between SZ and ZFP. *IEEE Trans. Parallel Distrib. Syst.*, 30(8):1857–1871. 78, 79

Taubman, D. S. and Marcellin, M. W. (2002). *JPEG2000: Image Compression Fundamentals, Standards and Practice*. Kluwer Academic. 9, 10

Thomas, G. (1981). *Principles of hydrocarbon reservoir simulation*. IHRDC, Boston, MA. 30

Thomas, L., Lumpkin, W., and Reheis, G. (1976). Reservoir simulation of variable bubble-point problems. *Society of Petroleum Engineers Journal*, 16(01):10–16. 84

Touma, C. and Gotsman, C. (1998). Triangle mesh compression. In *Proc. Graphics Interface Conf.*, pages 26–34, Vancouver, Canada. 33

Treib, M., Burger, K., Wu, J., and Westermann, R. (2015). Analyzing the effect of lossy compression on particle traces in turbulent vector fields. *In Proceedings of the 6th International Conference on Information Visualization Theory and Applications*. 78

Underwood, R., Di, S., Calhoun, J. C., and Cappello, F. (2020). Fraz: A generic high-fidelity fixed-ratio lossy compression framework for scientific floating-point data. 38, 78, 79, 80, 95

Wallace, G. K. (1992). The JPEG still picture compression standard. *IEEE Trans. Consum. Electron.*, 38(1):xviii–xxxiv. 10

Wang, J., Liu, T., Liu, Q., He, X., Luo, H., and He, W. (2019). Compression ratio modeling and estimation across error bounds for lossy compression. *IEEE Trans. Parallel Distrib. Syst.*, pages 1–1. 38, 104

Welton, B., Kimpe, D., Cope, J., Patrick, C. M., Iskra, K., and Ross, R. (2011). Improving i/o forwarding throughput with data compression. In *2011 IEEE International Conference on Cluster Computing*. IEEE. 78

Wilson, J. P. (2002). Compression of barotropic turbulence simulation data using wavelet-based lossy coding. In *Volume 1: Fora, Parts A and B*. ASME. 37

Witten, I. H., Neal, R. M., and Cleary, J. G. (1987). Arithmetic coding for data compression. *Commun. ACM*, 30(6):520–540. 33

Woodring, J., Mniszewski, S., Brislawn, C., DeMarle, D., and Ahrens, J. (2011). Revisiting wavelet compression for large-scale climate data using JPEG 2000 and ensuring data precision. In *Proc. IEEE Symp. Large Data Anal. Visual.* IEEE. 6, 37, 78

Yuan, Z., Hendrix, W., Son, S. W., Federrath, C., Agrawal, A., keng Liao, W., and Choudhary, A. (2016). Parallel implementation of lossy data compression for temporal data sets. In *2016 IEEE 23rd International Conference on High Performance Computing (HiPC)*. IEEE. 78

Zender, C. S. (2016). Bit grooming: statistically accurate precision-preserving quantization with compression, evaluated in the netCDF operators (NCO, v4.4.8). *Geoscientific Model Development*, 9(9):3199–3211. 36, 78

Zhang, J., Moon, A., Zhuo, X., and Son, S. W. (2019). Towards improving rate-distortion performance of transform-based lossy compression for HPC datasets. In *2019 IEEE High Performance Extreme Computing Conference (HPEC)*. IEEE. 78, 79, 81