Algorithme Multipoint Multi-Echelles

Dans ce chapitre nous commençons par une bibliographie sur les méthodes multi-échelles de modélisation de réservoir. Puis nous détaillons l'adaptation de l'algorithme précédent pour introduire plusieurs échelles. Après une brève comparaison des résultats avec l'algorithme mono-échelle, nous nous focalisons sur le point clé de la méthode qui est l'obtention d'une image d'entrainement grossière. Enfin nous testons les techniques décrites précédemment pour accélérer les simulations.

5.1. Etat de l'art

Comme expliqué précédemment, l'idée derrière le développement d'un algorithme multi-échelles est de modéliser l'architecture du réservoir et de prendre en compte les différentes échelles de sa structure et des données qui sont collectées. Le réservoir peut contenir de grandes hétérogénéités, comme des méandres, et des petites hétérogénéités provenant de processus tels que le dépôt ou l'accrétion. En outre, les données sont disponibles à différentes échelles. Il existe quatre sources d'information : les mesures aux puits, l'étude sismique 3D, les données dynamiques de réservoir et l'interprétation géologique de l'architecture. Les mesures aux puits sont les plus précises, mais les moins nombreuses. Au contraire, les données sismiques sont disponibles sur tout le réservoir mais leur niveau de précision est plus faible. Lors de la construction du modèle de réservoir, il faut prendre en compte le fait que chaque élément d'information a sa propre caractéristique et échelle. L'intégration des données sismiques a été beaucoup étudiée dans le cadre des méthodes géostatistiques à deux points : Deutsch et al. (1996) utilisent un cokrigeage, Behrens et Tran (1998) utilisent la simulation gaussienne séquentielle avec un krigeage par bloc, Lee et al. (2000) utilisent une approche hiérarchique basée sur les champs de Markov aléatoires (Markov Random Fields). Toutes ces approches font l'hypothèse d'une relation linéaire entre les échelles, ce qui n'est valable que pour des variables additives.

Au cours de la dernière décennie, les algorithmes multipoints ont connus un enthousiasme croissant de par leur habilité à modéliser des architectures géologiques complexes telles que les méandres. Pour capturer de façon efficace les grandes hétérogénéités du réservoir, c'est l'approche multi-grilles qui a été généralement utilisée (Strebelle, 2002 ; Straubhaar, *et al.*, 2011 ; Kolbjørnsen, *et al.*, 2014 ; Straubhaar et Malinverni, 2014 ; Strebelle et Cavelius, 2014). Le concept multi-grilles est un sous-échantillonnage de grilles, de la plus fine vers la plus grossière. Les sous-grilles et les template ne sont que des versions espacés de la grille et du template initiaux. S'il y a n_g sous-grilles, à la grille k, les pixels pris en compte sont espacés de 2^{ng-k} pixels. Dans ce schéma, les pixels d'une sous-grille grossière appartiennent aux sous-grilles de plus haute résolution. Cette approche multi-grilles est illustrée Figure 43 dans le cas de trois grilles. Les templates associés aux grilles sont aussi montrés dans le cas où leur taille est de 3×3 pixels.

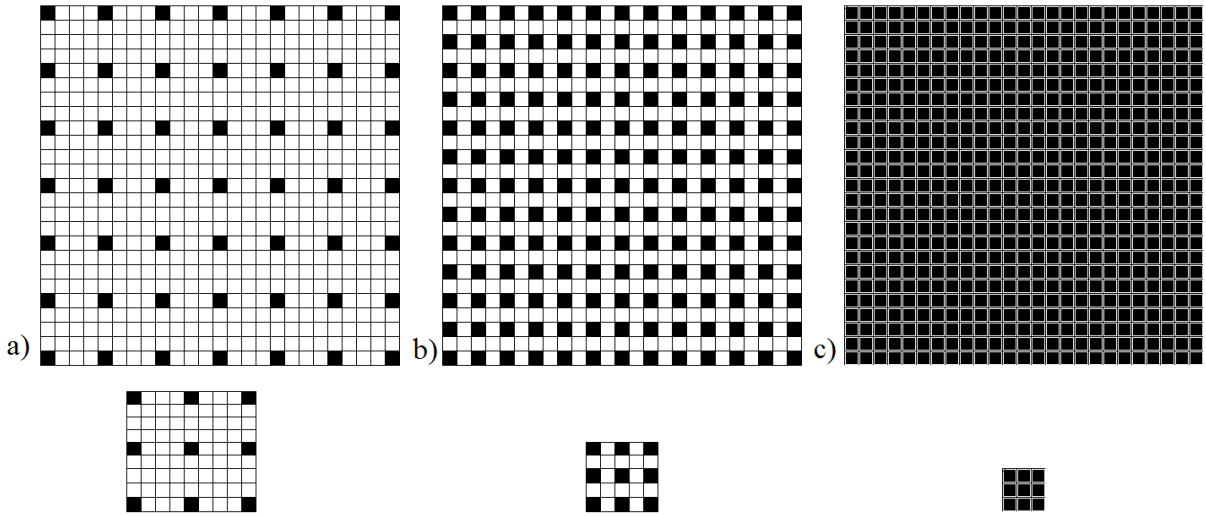

Figure 43 Illustration des sous-grilles et des sous-templates de l'approche multi-grilles. En noir ce sont les pixels qui appartiennent à la grille et qui sont utilisés. En blanc ce sont les pixels inutilisés.

Dans la même idée, certains auteurs se réfèrent à la simulation multi-échelles au lieu de simulation multi-grilles. Elle a été introduite en premier en infographie (De Bonet, 1997 ; Wei et Levoy, 2000 ; Ashikmin, 2001 ; Han, *et al.*, 2008). Au lieu de sous-échantillonner l'image d'entrainement, les valeurs de grille secondaire sont calculées à partir de celles de la grille de résolution plus fine. Ce calcul repose généralement sur l'interpolation bi-cubique en deux dimensions et sur l'interpolation tricubique en trois dimensions. En infographie, De Bonet (1997) a introduit l'idée de décomposer l'image d'entrainement en plusieurs sous-images de résolution inférieure, et d'utiliser les images de basse résolution pour contraindre la synthèse de l'image plus fine. Les techniques de Wei et Levoy (2000) pour l'infographie sont basées sur le même principe. L'image d'entrainement est décomposée en pyramide d'images de résolution décroissante, et l'image finale est simulée par résolution croissante. Le premier niveau de la réalisation finale est simulé avec le niveau le plus grossier de la pyramide des images d'entrainement. Lors de la simulation d'un niveau plus fin de la réalisation finale, on tient compte de la grille de simulation précédente. Un autre travail remarquable sur la synthèse de texture multi-échelles est la synthèse de l'image du continent, avec zoom de Han *et al.* (2008). La variété des zooms exigerait en entrée de simulation une quantité impossible d'images de différentes résolutions. Dans leurs travaux, l'image est construite à partir d'un graphe d'exemples de différentes images de résolution liées par des liens pondérés, réfléchis et orientés. En raison de l'approche multi-échelle, la simulation prend en compte les différentes échelles lors du choix des candidats. Il ne nécessite que quelques exemples en entrée.

Le principal avantage de ces approches est l'effet de dilatation du voisinage. Un petit template dans un schéma multi-échelles apporte plus ou moins autant d'information qu'un gros template dans un schéma mono-échelle. Cela permet d'améliorer la capture de grosses hétérogénéités avec de plus petits templates, ce qui rend la simulation plus efficace en termes de temps de calcul.

Le potentiel de l'approche multi-échelles, décrite en infographie, commence à être étudié en géosciences depuis peu.

Tahmasebi et Caers (2014) ont développé une extension de CCSIM dans une version multi-échelles appelée MS-CCSIM. L'image d'entrainement est décomposée en une pyramide d'images de résolution décroissante à chaque niveau. Cette pyramide d'images est utilisée pour réduire la taille de l'aire scannée dans l'image d'entrainement. Le domaine de la recherche à

l'échelle grossière est réutilisé à une échelle plus fine. Le domaine à l'échelle plus fine est centré sur le domaine à l'échelle plus grossière.

Un workflow intéressant a été proposé par Li et Caers (2011) pour un modèle hiérarchique de réservoir chenalisé. La première étape consiste à modéliser les grandes structures architecturales du réservoir à partir des données sismiques (emplacements des chenaux), puis à l'aide d'une technique multipoints, ils simulent les nappes de schiste le long des surfaces de ces chenaux. La deuxième étape de simulation consiste à ajouter des chenaux individuels dans les grands emplacements précédents, puis à re-simuler des nappes de schiste le long de ces plus petits chenaux. La dernière étape consiste à simuler les lithofaciès avec une simulation séquentielle gaussienne dans les petits chenaux.

L'approche multi-échelle est également bien connue dans la simulation basée sur les ondelettes. En infographie, Crouse et al. (1998) ont étudié les dépendances de coefficients d'ondelettes à différentes échelles pour la modélisation statistique des images. Des études montrent que la transformée en ondelettes discrètes (DWT) peut identifier les échelles dominantes et donner les relations entre les échelles (Aradhye et al, 2003 ; Droujinine, 2006), mais aussi gérer l'intégration multi-échelles de données (Gloaguen et Dimitrakopoulos, 2009). En 2012, Chaterjee *et al*. utilisent l'analyse par ondelettes dans la simulation par patch afin de réduire la dimension de la base de données des motifs.

Nous proposons ici, un algorithme multi-échelles et multi-résolutions qui est l'extension de l'algorithme précédemment étudié chapitre 4.

Dans la première partie, nous détaillons étape par étape l'algorithme multi-échelles. Dans la deuxième partie, une comparaison des algorithmes mono et multi-échelles montre l'amélioration apportée par l'introduction d'une seconde échelle. La troisième partie est consacrée à la représentation de l'image d'apprentissage à l'échelle grossière. Enfin, dans la quatrième partie, nous testons les deux techniques précédemment employées pour accélérer l'algorithme et comparer les résultats en temps CPU ainsi que la qualité de l'image finale synthétisée.

5.2. Algorithme

L'algorithme décrit dans le chapitre précédent atteint ses limites lorsque les hétérogénéités sont de grande taille. Il faut alors prendre de grands templates pour pouvoir capturer ce hétérogénéités ce qui implique un temps de calcul conséquent et des résultats assez décevants en terme de qualité de reproduction de l'image d'entrainement. Le but, ici, est de créer une échelle grossière, d'une résolution plus faible, donc rapide à synthétiser et qui rend compte des tendances des hétérogénéités. Dans un deuxième temps, on passe à l'échelle fine pour ajouter des détails à l'image simulée au préalable à l'échelle grossière. La synthèse à l'échelle fine ne nécessite alors pas de grands voisinages puisque que la forme générale des grandes hétérogénéités a été capturée au préalable dans l'image grossière. La synthèse est donc plus rapide.

L'algorithme de synthèse multi-échelles se déroule comme suit. Premièrement il faut définir les échelles selon les différents niveaux de résolution souhaités ou imposés, du plus bas au plus élevé. Il faut autant d'images d'entrainement que d'échelles. Deuxièmement on simule les réalisations de la plus grossière à la plus fine, en tenant compte à chaque niveau de tous les niveaux inférieurs. Ici, l'algorithme est expliqué sur deux échelles, pour simplifier. Il y a donc deux images d'entrainement, l'une pour l'échelle grossière, et l'autre pour l'échelle fine.

La première étape de l'algorithme consiste à synthétiser l'image grossière à partir de l'image d'entrainement grossière à partir de l'algorithme mono-échelle. Une fois la simulation faite, cette image ne change plus. La deuxième étape consiste à synthétiser l'image fine à partir de l'image d'entrainement fine en tenant compte de l'image grossière que l'on vient de synthétiser. Cette méthode est décrite ci-dessous pour la simulation sans contrainte.

Le lien entre les deux échelles est fait par la structure du template utilisée pour simuler la grille fine. Ce template contient les deux templates fins et grossiers, il contient donc dans une première partie les voisins de la grille fine et dans une deuxième partie les voisins de la grille grossière. Ils sont stockés comme une paire, comme le « dual-template » introduit par Arpat et Caers (2007) pour leur simulation multigrille. La recherche du motif le plus proche sur la grille fine est guidée par les deux échelles en comparant simultanément le template dans les deux images d'apprentissage.

Dans cette deuxième simulation, la première étape est d'initialiser la grille fine en sélectionnant un patch à coller au centre, mais pas au hasard: le choix du motif est contraint par le patch localisé au même endroit sur la grille grossière. Ensuite, nous remplissons la grille fine, couronne par couronne jusqu'à ce que la grille soit entièrement simulée. Une couronne est définie à un stade donné et comprend les mailles de grille qui n'ont pas déjà été simulées, et qui entourent immédiatement les mailles déjà simulées. La première couronne est celle autour du motif initial collé au centre. Une fois que cette première couronne est simulée, nous construisons la seconde autour de la première et ainsi de suite jusqu'à ce que la grille soit entièrement simulée. Pour toutes les mailles d'une couronne donnée, on compte le nombre de voisins contenant une valeur. Puis, on visite ces mailles en partant de celles qui a le plus de voisins simulés à celle qui en a le moins.

L'algorithme est assez proche de celui présenté plus haut pour l'approche mono-échelle. Les différences apparaissent dans la construction de la base de motifs et dans la comparaison des voisinages.

Seule la simulation à l'échelle fine est détaillée ici. On rappelle que I_c désigne l'image grossière synthétisée en utilisant l'algorithme mono-échelle à partir de l'image d'apprentissage Ti_c.
I_f est la réalisation fine à simuler à la fois à partir de l'image d'entrainement fine Ti_f, de l'image d'entrainement grossière Ti_c et de l'image déjà simulée I_c. p_f est le patch de pixels à simuler sur I_f à l'itération en cours, il est centré sur la maille fine i.
w est la fenêtre de contexte. $w(i)$ contient à la fois les voisins sur la grille fine $wf(i)$ et les voisins sur la grille grossière $wc(i)$. La fenêtre fine est centrée sur la maille i et la fenêtre grossière est centrée sur la maille grossière qui contient la maille i. Les étapes successives de la deuxième simulation de l'algorithme sont les suivantes.

Étapes de l'algorithme:

- Initialisation de I_f, guidée par l'image grossière déjà simulée I_c. La fenêtre de contexte ne contient que les voisins à l'échelle grossière. Ainsi, le motif choisi a sa partie de résolution grossière correspondant aux voisins sur la grille grossière au centre de I_c.
- Jusqu'à ce que toutes les mailles sur la grille fine soient visitées une fois:
    - Compte tenu des mailles déjà simulées sur la grille fine, identifier la couronne actuelle comme l'ensemble des mailles ayant déjà au moins un voisin simulé.
    - Trier les mailles de la couronne de celles qui a le plus de voisins à celles qui en a le moins. (Seuls les voisins sur la grille fine sont pris en compte, car sur la grille

grossière toutes les mailles sont simulées. Donc les mailles de la grille fine ont toutes le même nombre de voisins sur la grille grossière)

- On boucle sur les mailles selon le classement établi
    - Identifier le voisinage *w(i)* de la maille courante *i* selon le template. Il contient une partie fine avec les voisins sur la grille fine, et une partie grossière avec les mailles grossières correspondantes.
    - Comparer *w(i)* à la bibliothèque de motifs de référence recueillis dans les images d'entrainement et choisir celui qui est le plus proche de *w(i)*, la distance *d* entre les motifs et *w(i)* étant définie à partir d'une métrique. Notre choix par défaut est la norme L_2 pondérée par des poids en fonction de la position des mailles par rapport à la maille centrale (voir Figure 33).
    - Le patch central est alors extrait de la partie fine du motif sélectionné et collé sur *I* dans le groupe de pixels correspondant centré sur *i*.

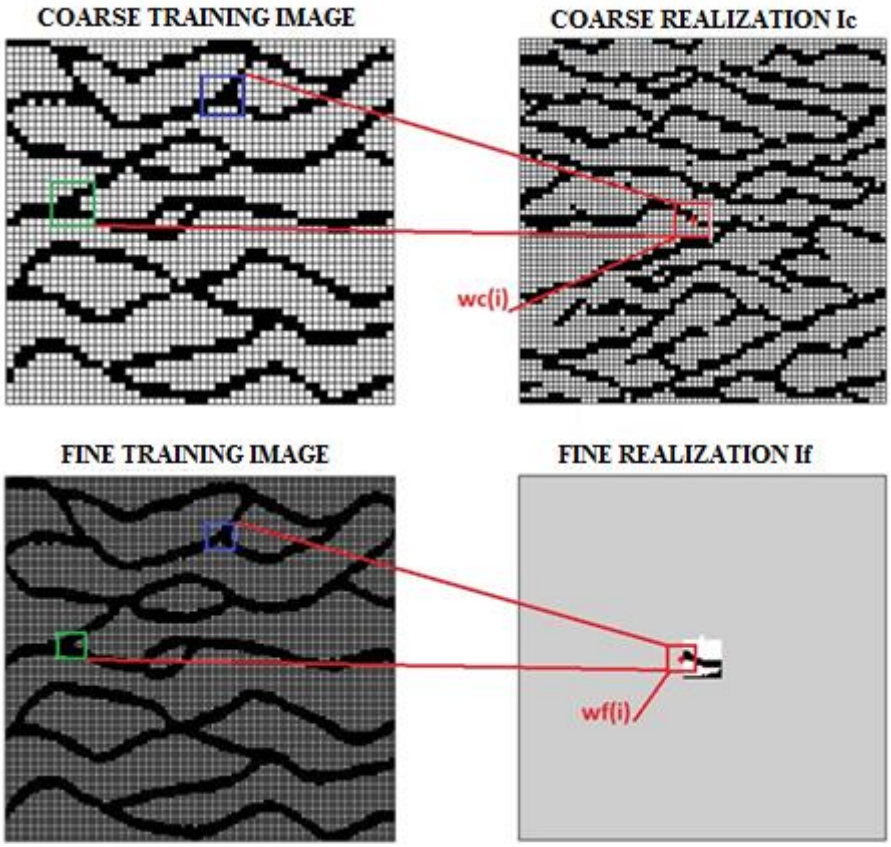

Figure 44 En haut à gauche, image d'entrainement grossière TI_c. En haut à droite, image synthétisée grossière I_c. En bas à gauche, image d'entrainement fine TI_f. En bas à droite, image synthétisée fine I_f. En bas à droite, pour le pixel *i* sur I_f, le pixel équivalent sur I_c est également marqué +, donc on compare le template en rouge sur la droite à la fois avec les motifs de la grille grossière et avec les motifs de la grille fine sur la gauche (vert et bleu).

Jusqu'ici on scanne toute l'image d'entrainement lorsque l'on simule un pixel. D'autres stratégies sont envisagées dans la section 5.5.

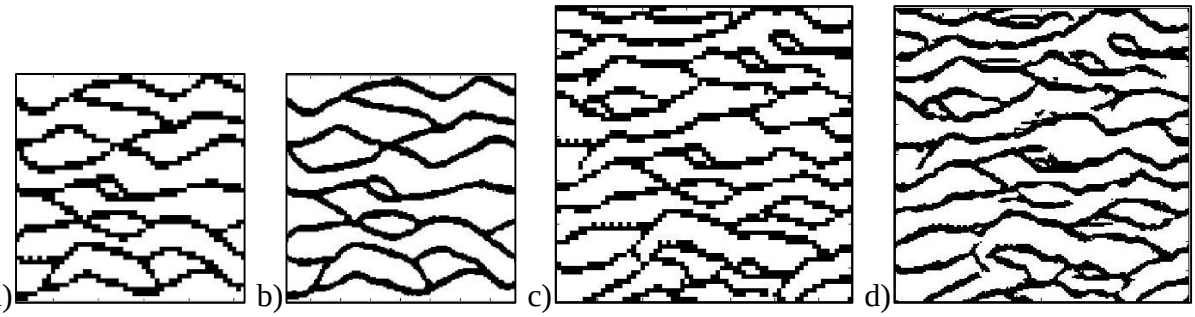

Figure 45 Simulation multi-échelles avec deux échelles. a) l'image d'entrainement grossière. b) l'image d'entrainement fine. c) Réalisation à l'échelle grossière. d) Réalisation à l'échelle fine. L'image d'entrainement grossière est obtenue à partir de l'image d'entrainement fine en attribuant à la maille grossière la valeur de la médiane du bloc de mailles fines correspondantes. Le template utilisé pour simuler la réalisation à l'échelle grossière comprend 9 × 9 mailles. Pour la simulation à l'échelle fine, la partie grossière du template comprend 3 × 3 mailles et la partie fine du template comprend 9 × 9 mailles.

Un exemple est illustré sur la Figure 45. Dans ce cas, l'image d'entrainement à l'échelle grossière est obtenue à partir de l'image d'entrainement à l'échelle fine. Pour chaque maille grossière, on calcule la médiane des mailles fine à l'intérieur. L'image d'entrainement grossière est constituée de 53 × 53 mailles. La réalisation à l'échelle grossière est obtenue en utilisant un template 9 × 9 et un patch 3 × 3. Il comprend 67 × 67 mailles. Dans ces conditions, le temps de simulation est inférieur à une seconde. La réalisation à l'échelle grossière semble plutôt bonne : les chenaux sont connectés et continus. Ensuite, on simule la réalisation fine (Figure 45d) conditionnellement à la réalisation à l'échelle grossière (Figure 45c). Le dual-template est construit à partir d'un template fin de 9 × 9 mailles, et d'un template grossier de 3 × 3 mailles. La réalisation fine respecte la tendance donnée par la réalisation grossière et reproduit les objets décrits par l'image d'entrainement fine. L'avantage de ce procédé multi-échelles est que l'on utilise un template de taille réduite pour reproduire l'image d'entrainement. Comme le template est petit, le temps de calcul diminue de manière significative par rapport à l'algorithme mono-échelle. Si l'on compare les réalisation représentées sur la Figure 29d et la Figure 45d, le temps de calcul est divisé par 7.

5.3. Comparaison des résultats à une et deux échelles

Tout d'abord, l'algorithme multi-échelles améliore la représentation des grandes structures. En effet, la synthèse multi-échelles permet de représenter de grandes hétérogénéités à l'échelle grossière avec un template plus petit (car la résolution est plus basse) et les détails sont ajoutés à l'échelle fine avec de même un petit template (car les grandes hétérogénéités sont déjà en place).

L'échelle grossière est une échelle de faible résolution, les images d'entrainement grossières sont (ici) 9 fois plus petites que l'image d'entrainement fine. Un pixel à l'échelle grossière représente les valeurs du bloc de 9 pixels adjacents à l'échelle fine. Les grandes hétérogénéités peuvent être capturées avec un template plus petit parce que l'image est plus petite. Les améliorations sont clairement visibles sur des structures telles que des rivières ou des matrices granulaires (voir Figure 46).

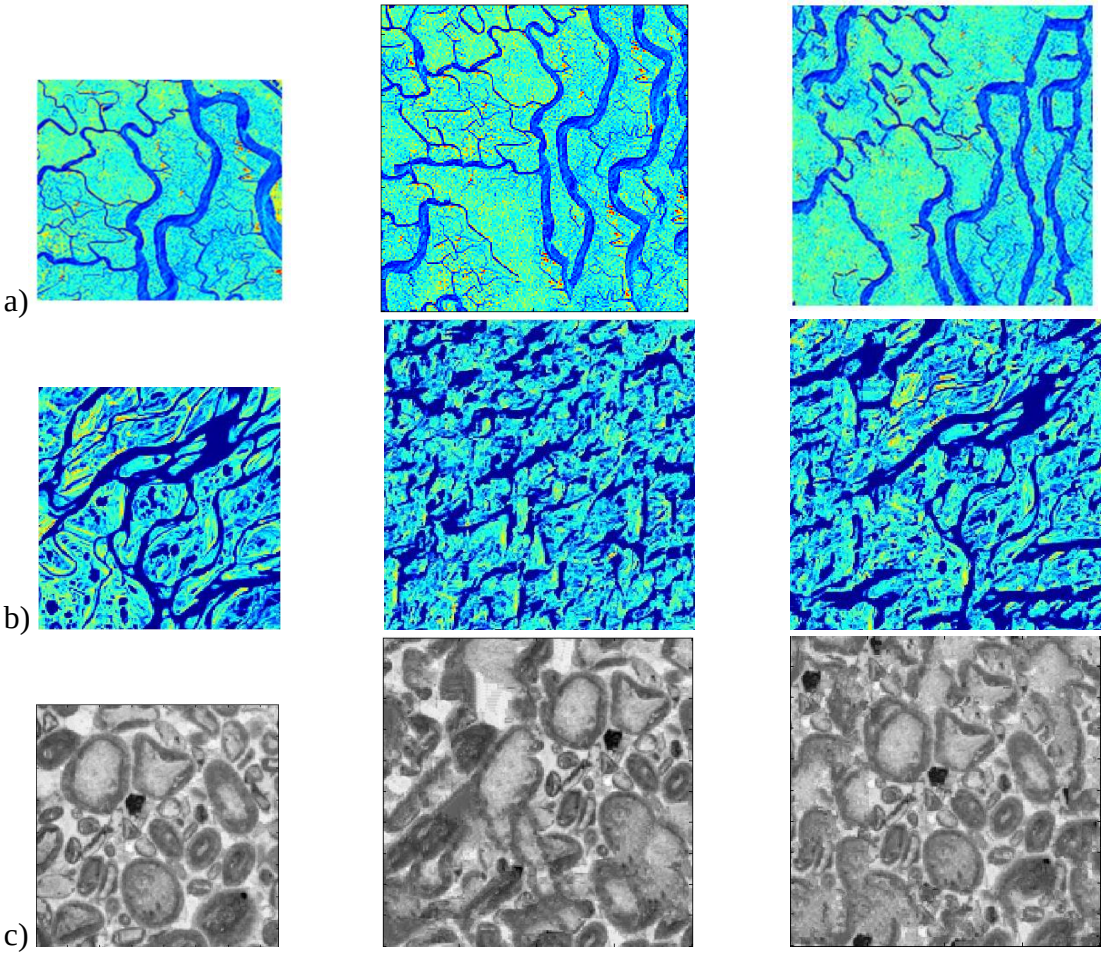

Figure 46 Comparaison de l'algorithme mono-échelle avec l'algorithme à deux échelles. Dans les colonnes de gauche à droite: Image d'entrainement ; Réalisation mono-échelle ; Réalisation multi-échelles. a & b) Simulation mono-échelle : Taille de template 21 × 21 et Taille de patch 3 × 3. Simulation deux échelles : Taille du template grossier 15 × 15 et Taille de patch 3 × 3, Taille du template fin 9 x 9, Taille du template grossier associé 3 x 3, et taille du patch 3 × 3. c) Simulation mono-échelle : Taille de template 21 × 21 et taille de patch 3 × 3. Simulation à deux échelles : taille du template grossier 15 × 15 et taille du patch 3 × 3, Taille du template fin 15 × 15, taille du template grossier associé 5 x 5, et taille du patch 3 × 3.

La Figure 46 montre les améliorations apportées par une approche multi-échelle. Dans la première colonne on trouve les images d'entrainement. Dans la deuxième colonne, on trouve les réalisations générées avec l'algorithme mono-échelle (chapitre 4). Dans la troisième colonne, on trouve les images construites par la méthode multi-échelles. Les deux premières images (lignes a et b) sont des photos satellites des rivières avec de grands méandres, la troisième image est une photo d'un morceau de quartz sous microscope.
On voit que dans la deuxième colonne, les réalisations de l'algorithme à une échelle ressemblent aux images d'entrainement mais présente des artefacts comme des lignes horizontales (a. 2ème colonne) ou diagonales (c. 2ème colonne). Pour diminuer le temps de calcul et améliorer la représentation des images on introduit une deuxième échelle de résolution plus basse. La troisième colonne montre les réalisations fines du processus multi-échelle. Les chenaux sont visibles et certains sont reliés. Clairement les réalisations finales semblent provenir d'un même processus de formation que les images d'apprentissage.

D'autre part, cet algorithme améliore également le temps de calcul. En comparaison avec la simulation à une échelle, dans laquelle la taille du template est de 21 × 21, la simulation à deux échelles utilise pour la simulation grossière un template de taille de 9 × 9 pixels, et pour la simulation fine un template fin de 9 × 9 et un template grossier associé de 3 × 3. Le temps de

calcul est divisé par 7. La simulation à l'échelle grossière tourne en une ou deux secondes, et la simulation fine échelle en 68 secondes.

Table 6 Comparaison des temps de simulation entre l'algorithme mono-échelle et l'algorithme multi-échelles

	Figure 46 a)	Figure 46 b)	Figure 46 c)
Simulation mono-échelle	T_0 = 420 sec	T_0	T_1
Simulation à 2 échelles	T_0 /6	T_0 /6	T_1/3.6

Le point intéressant de cet algorithme est que la réalisation fine est conditionnée par la réalisation à l'échelle grossière. Comme on peut le voir sur la Figure 47, l'image fine suit les tendances données par l'image grossière. Les images grossières donnent l'emplacement des méandres, ainsi que leur forme. La simulation fine vient ajouter des détails, sans modifier les grandes structures de l'image.

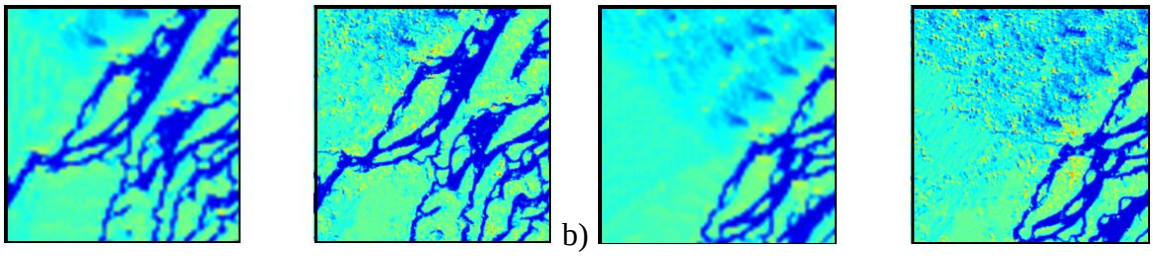

Figure 47 Impact de la réalisation à l'échelle grossière sur la réalisation à l'échelle fine. Sur les figures a) et b) à gauche se trouvent les réalisations à l'échelle grossière et à droite les réalisations à l'échelle fine. Les réalisations à l'échelle grossière sont simulées avec un template de taille 13×13 et un patch de 3×3 pixels. Les réalisations à l'échelle fine sont simulées avec l'algorithme multi-échelle, un template de taille 15×15 et un patch de taille 5×5.

5.4. Origine de l'image grossière

Nous nous concentrons ici sur deux images décrivant la même structure, mais à des échelles différentes. Par exemple, à l'échelle grossière les chenaux apparaissent mais pas les différentes couches de dépôts de sédiments et à l'échelle fine tout est visible.

5.4.1. Cas d'une propriété continue

Cette section est consacrée au calcul de l'image à l'échelle grossière à partir de l'image à l'échelle fine. On considère une propriété continue telle que la perméabilité. Les images grossières viennent des images de résolution fine filtrées. Nous avons testé différents types de filtrage: moyenne, médiane, et NL-means. Nous montrons leur impact sur l'image de synthèse.

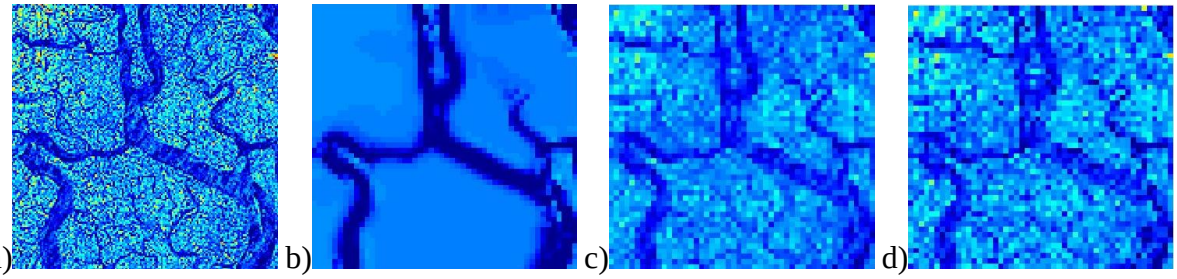

Figure 48 Comparaison des différentes images secondaires en fonction de leur voie d'acquisition. a) l'image d'origine. b) l'image filtrée avec NL-means, puis la moyenne. c) l'image moyennée. d) l'image après la prise de la médiane.

5.4.1.1 Moyenne et Médiane

L'idée est que l'image à l'échelle grossière est la même que l'image à l'échelle fine, mais à une résolution inférieure. La méthode de la moyenne (médiane) consiste simplement à prendre la moyenne (médiane) d'un bloc de pixels de la grille fine et à l'affecter au pixel central de la grille et à éliminer les autres pixels du bloc.
L'équation ci-dessous donne la valeur du pixel $TI_c(i, j)$, appartenant à l'image grossière, comme étant la moyenne des pixels de $TI_f(p, q)$, appartenant à l'image fine.
Soient NX_f et NY_f, les nombres de pixels suivant les axes x et y pour l'image fine, et NX_c et NY_c, les nombres de pixels suivant les axes x et y pour l'image grossière. a est la différence de résolution entre l'image fine et l'image grossière. Alors, on a :

$$NX_c = {NX_f}/{a} \qquad et \qquad NY_c = {NY_f}/{a}$$

$$\forall i \in [1, NY_c], \forall j \in [1, NX_c], \qquad TI_c(i,j) = \frac{1}{a^2} \sum_{p=a\times(i-1)}^{a\times i} \sum_{q=a\times(j-1)}^{a\times j} TI_f(p,q) \qquad \textbf{Equation 23}$$

Par exemple ici dans la Figure 48, a = 3, un seul bloc de grille sur neuf est maintenu et sa valeur est la moyenne des neuf pixels (Figure 48 c) ou la médiane des neuf pixels (Figure 48 d).

5.4.1.2 NL-means

L'algorithme Non Local means est un algorithme de traitement d'image pour débruiter des images. Contrairement aux filtres de lissage, cet algorithme ne met pas à jour la valeur d'un pixel selon une moyenne du voisinage mais plutôt en le moyennant sur l'ensemble des pixels de l'image qui ont le même voisinage. Chaque pixel est pondéré selon la distance entre son voisinage et celui du pixel mis à jour. L'algorithme NL-means a été publié par Antoni Buades *et al*. en 2005.

Considérant l'image *I*, le pixel *i* à simuler, et *v(i)* sa valeur, *NL[v](i)* est la valeur estimée de *v(i)*, donnée par :

$$NL[v](i) = \sum_{j \in I} w(i,j)v(j)$$

$$0 \le w(i,j) \le 1 \quad et \quad \sum_j w(i,j) = 1 \qquad \textbf{Equation 24}$$

La famille de poids $\{w(i,j)\}_j$ dépend du degré de similitude entre les pixels *i* et *j*. La similarité entre deux pixels *i* et *j* dépend de l'intensité des vecteurs de niveau de gris *v(Ni)* et *v(Nj)* où *Nk* est un voisinage carré de taille fixe et centré sur le pixel *k*. La similitude est mesurée avec une norme L_2 et les poids sont donnés par :

$$w(i,j) = \frac{1}{Z(i)} e^{\left[-\frac{\|v(Ni)-v(Nj)\|^2_{2,a}}{h^2}\right]} \qquad with \;\; Z(i) = \sum_j e^{\left[-\frac{\|v(Ni)-v(Nj)\|^2_{2,a}}{h^2}\right]}$$

h est le degré de filtrage. Il contrôle la décroissance de la fonction exponentielle, et donc la décroissance des poids.
L'algorithme a deux autres paramètres tout aussi importants qui pilotent le débruitage : *t* et *f*.

t est la zone scannée pour la recherche des voisinages similaires. *F* est la taille du voisinage *Nk* centrée sur *k*. Ils sont représentés sur la Figure 49 :

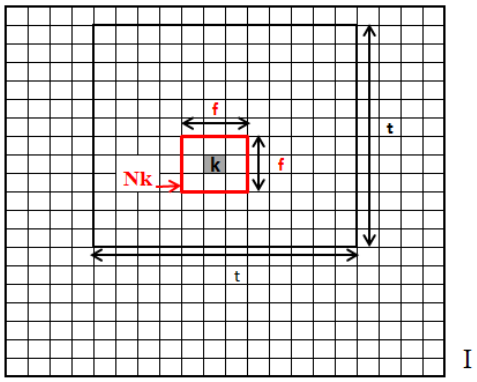

Figure 49 Illustration des paramètres de l'algorithme NL-means

La Figure 50 illustre la sensibilité au paramètre *h*, de l'image filtrée. Plus *h* augmente plus le filtrage est efficace et les plus petits détails sont effacés. Toutefois, avec un *h* supérieur à 100, l'image est trop floue, même la structure principale commence à être effacée.

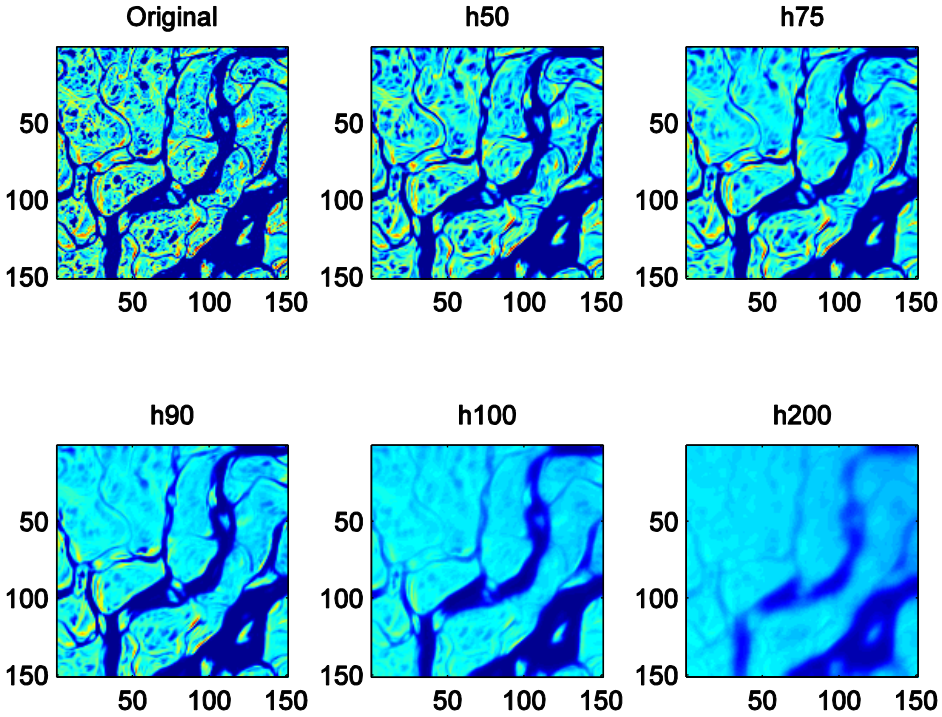

Figure 50 Photo satellite du delta du fleuve Lena. Comparaison de l'impact du degré de filtrage *h* sur le processus de débruitage. Les autres paramètres sont fixés : *t* est égal à 50 et *f* est égal à 7. En haut, de gauche à droite : La photo originale, la photo débruitée avec *h* égal à 50 et avec *h* égal à 75. En bas, de gauche à droite : images débruitées avec *h* égal respectivement à 90, 100 et 200.

5.4.1.3 Impact du filtrage sur l'image finale

Intuitivement, il semble normal qu'une image avec un contraste plus élevé donne de meilleurs résultats, quelle que soit la manière de calculer la distance entre deux motifs. Pour tester et vérifier cette idée, nous avons filtré une photo satellite bruitée montrant le delta du Gange (Figure 51 a) pour obtenir une image de contraste élevé (Figure 51 c). Ensuite, nous

procédons à la simulation mono-échelle avec les deux images d'entrainement différentes (filtrée et non filtrée). Les résultats (Figure 51 b et d) montrent que l'image synthétisée à partir de la photo filtrée donne une meilleure qualité dans le rendu final.

a) b) c) d)

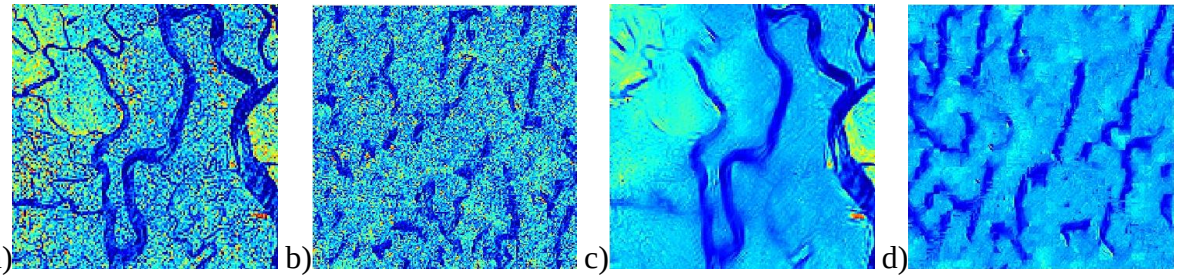

Figure 51 Influence du filtrage sur la simulation à une échelle. a) l'image d'entrainement originale b) l'image simulée à partir de l'image non-filtrée c) l'image d'entrainement filtrée d) l'image synthétisée à partir d'une image d'entrainement filtrée. Les deux réalisations sont simulées avec un voisinage de 21 x 21 et un patch de 3 x 3.

Nous avons montré précédemment que la réalisation à l'échelle grossière guide la réalisation à l'échelle fine. Donc une réalisation à l'échelle grossière qui représente bien les chenaux (leur structure, leur espacement, *etc.*), permet d'obtenir une bonne réalisation à l'échelle fine. Comme on vient de le montrer sur la **Figure 51**, à partir d'une image de faible contraste on ne parvient pas à recréer de chenaux, donc introduire une phase de filtrage à l'échelle grossière permet d'améliorer la qualité de l'image d'entrainement grossière et par conséquent d'améliorer le rendu de la simulation à l'échelle grossière.

a) b) c) d)

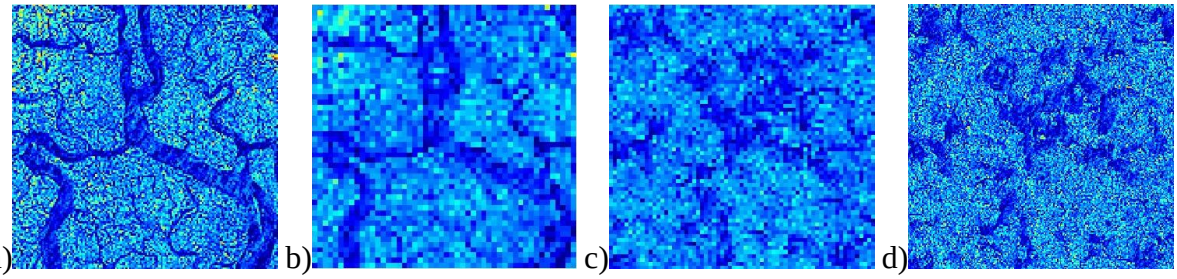

Figure 52 Impact du non-filtrage sur la simulation à deux échelles. a) l'image d'entrainement fine. b) l'image d'entrainement grossière, obtenue par moyenne de l'image fine c) la réalisation à l'échelle grossière avec un template de 9 x 9 et un patch de 3 × 3. d) la réalisation fine avec un template fin de 9 × 9 et un template grossier associé de 3 × 3 et un patch de 3 × 3

a) b) c) d)

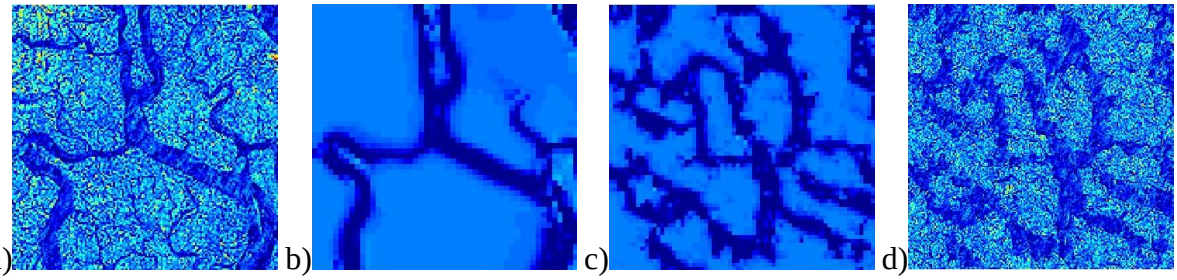

Figure 53 Impact du filtrage sur la simulation à deux échelles a) l'image d'entrainement fine. b) l'image d'entrainement grossière, obtenue par NL-means de l'image fine, $h = 100$, $f = 11$, $t = 50$. c) la réalisation à l'échelle grossière avec un template de 9 x 9 et un patch de 3 × 3. d) la réalisation fine avec un template de 9 × 9 et un template grossier associé de 3 × 3 et un patch de 3 × 3

Au cours du processus de simulation multi-échelles, la simulation mono-échelle à l'échelle grossière est essentielle, en ce qu'elle donne la tendance, la connectivité et la continuité

de grandes structures. Si la réalisation grossière n'est pas de bonne qualité, il en va de même pour la réalisation fine. L'algorithme mono-échelle est plus performant avec l'image d’entrainement de haut contraste.
Sur la Figure 52, l’image d’entrainement (Figure 52 a) a un faible contraste entre les chenaux de la rivière et la terre. L'image d’entrainement grossière (Figure 52 b) est obtenue simplement par moyenne de l’image fine. Donc elle a encore un faible contraste. La réalisation à l'échelle grossière (Figure 52 c) ne représente pas correctement la continuité des chenaux, ce qui conduit à la mauvaise qualité de la réalisation fine (Figure 52d).
Dans la Figure 53 a) nous gardons la même image fine d’entrainement, mais nous l’avons filtrée à l’aide de NL-means pour obtenir l'image d’entrainement grossière (Figure 53 b) plus contrastée. La réalisation grossière Figure 53c) est de meilleure qualité, les chenaux sont continus et connectés. La réalisation fine (Figure 53 d) suit les tendances données par la réalisation grossière, et représente mieux les caractéristiques de l’image d’entrainement fine.

5.4.2. Cas d’une propriété discrète

Dans cette section on se concentre sur les images représentant des propriétés discrètes, telles que des fractures. Ces images peuvent être très complexes, contenant plusieurs niveaux de fracturations et donc de fractures de tailles différentes. Comme les fractures impactent très fortement les écoulements à l’intérieur du réservoir, il est nécessaire de reproduire au mieux le réseau naturel.
L’image d’entrainement utilisée Figure 54 vient d’une carte de fractures dans une formation calcaire du champ de Karahayit dans l’ouest de la Turquie (Jafari et Babadagli, 2010). Sur l’image les fractures sont en noir (ce sont des zéros dans l’image), et la matrice encaissante en blanc (représentée par des 1).

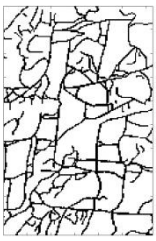

Figure 54 Image d’entrainement fine (125×192 pixels).

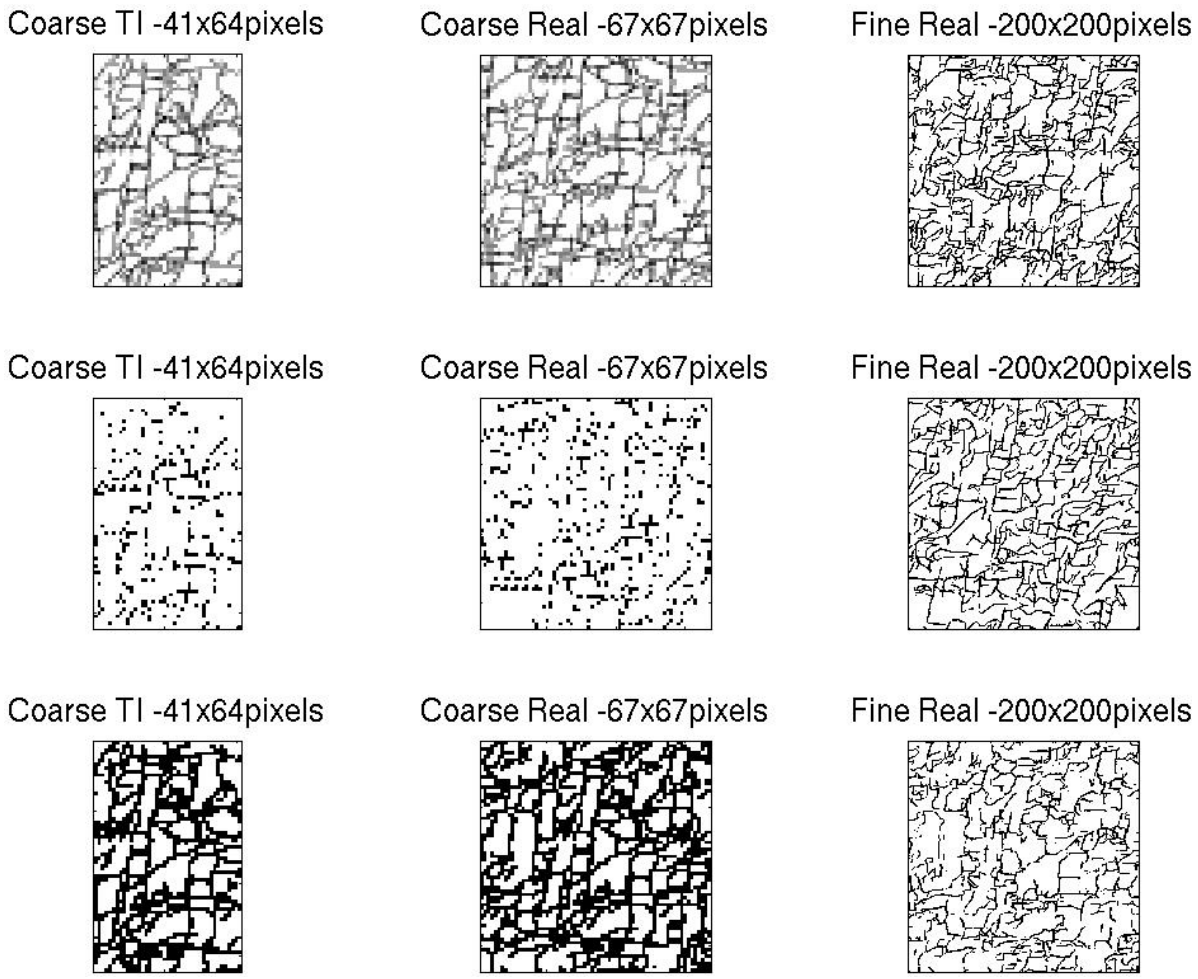

Figure 55 Impact de la technique utilisée pour obtenir l'image d'entrainement grossière. 1ère ligne : moyenne arithmétique. 2ème ligne : médiane. 3ème ligne : minimum. 1ère colonne : Les images d'entrainement grossières. 2ème colonne : Les réalisations grossières simulées à partir des images grossières de la 1ère colonne. 3ème colonne : Réalisations fines conditionnées par les réalisations grossières de la 2ème colonne. (Paramètres utilisés : différence de résolution entre les échelles 3×2 ; taille de template grossier 9×9 ; taille du patch 3×3 ; taille du dual template 13×13 fin, 3×3 grossier ; taille du patch 3×3 ; 3 classes)

Trois techniques ont été testées ici : la moyenne et la médiane, qui sont identiques à celles expliquées section 5.4.1 et la prise du minimum, qui est simplement le fait de prendre une occurrence de fracture sur le pixel grossier si le bloc de pixels fins est traversé par une fracture. La moyenne donne une image grossière avec des variations continues le long des fractures. La médiane donne plutôt un ensemble de points. Et la prise du minimum donne une version discrète de l'image obtenue par la moyenne (Figure 55, 1ère colonne).
Intuitivement on s'attend à ce que les prises de moyenne et du minimum donnent de meilleurs résultats car les images d'entrainement grossières obtenues ressemblent plus à celle d'origine et décrivent mieux le réseau de fractures. Au contraire, l'image d'entrainement grossière obtenue par la prise de médiane, ne ressemble pas du tout à l'image d'origine mais plutôt à un ensemble presque aléatoire de points. La 2ème colonne (Figure 55) semble confirmer cette intuition. Les réalisations grossières sont semblables aux images d'entrainement grossières à partir desquelles elles ont été simulées. Cependant les résultats de la 3ème colonne (Figure 55) inversent ces observations. La réalisation fine conditionnée par la réalisation grossière obtenue par moyenne, présente une densité de petites fractures trop élevée. Celle conditionnée par l'image grossière obtenue par prise de minimum, ne présente pas assez de connectivité entre

les fractures. Finalement c'est la réalisation fine conditionnée par la réalisation grossière obtenue par prise de médiane qui donne les meilleurs résultats.

5.5. Techniques de diminution du temps de calcul

On teste ici les méthodes testées en section 4.4 pour diminuer le temps de calcul.

5.5.1. Scan partiel de l'image d'entrainement fine

La première simulation mono-échelle à l'échelle grossière reste la même. Le temps de la simulation grossière mono-échelle est très court, et l'image d'entrainement grossière est petite. Nous n'avons donc pas besoin de réduire la partie scannée. On ne joue que sur la deuxième simulation, celle de la réalisation fine à partir de la réalisation grossière et de l'image d'entrainement fine.
On se concentre ici sur la simulation de la réalisation à l'échelle fine. Pour attribuer une valeur à un pixel de la grille fine, on ne scanne qu'une partie de la base de données des doubles motifs (les motifs qui comprennent une partie fine et une partie grossière).
Contrairement à l'algorithme mono-échelle, ici la partie scannée peut être réduite à 50% sans trop dégrader l'image finale. Une des raisons principales venant du fait que la simulation est contrainte par la réalisation à l'échelle grossière qui donne la continuité et la connectivité des chenaux.

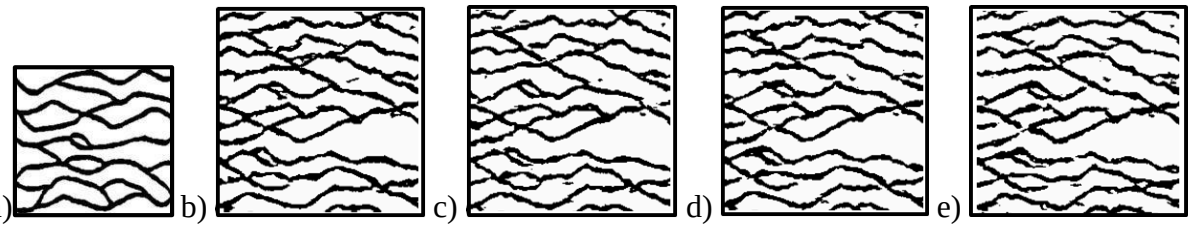

Figure 56 Impact de la proportion de l'image d'entrainement scannée au cours de la simulation a) l'image d'entrainement b) 100% c) 75% d) 50% e) 25%. Réalisation grossière simulée avec un template 9×9 et un patch 3×3. Réalisations fine simulées avec un dual-template 9×9, 3×3 et un patch 3×3.

Table 7 Comparaison du temps CPU de simulation avec des proportions différentes image numérisée de la formation

Proportion scannée	100%	75%	50%	25%
Temps de simulation	T_2=68sec	0.76 T_2	0.48 T_2	0.23 T_2

Sur la Figure 56, on peut voir l'effet de la proportion de l'image scannée au cours du processus de simulation. Ces images sont construites à partir de la même réalisation grossière. Elles sont simulées avec un template de 9 × 9 et un patch 3 × 3. Ensuite, lors de la simulation de la réalisation fine, nous scannons seulement une certaine proportion de la base de données de dual-templates pour trouver le meilleur motif. Cette proportion passe de 100% à 25% (Figure 56 b à e). Dans la Table 7, les temps de simulation relatifs sont donnés pour chaque simulation. Le temps de référence T_2 est celui de la simulation avec le scan total de l'image d'entrainement. Les temps sont proportionnels à la partie de l'image scannée. Le temps est divisé par deux lorsque la moitié de l'image d'apprentissage est balayée.

5.5.2. Kmeans clustering

La base de données est ici construite avec les motifs extraits des images d'entrainement à la fois grossière et fine. On rappelle qu'un template multi-échelles contient d'abord le template extrait de l'image d'entrainement fine et un deuxième template extrait de l'image d'entrainement grossière correspondant au template fin.
Nous regroupons ces templates multi-échelles en *k* classes et chaque classe est représentée par le template multi-échelle le plus proche de la moyenne des templates de la classe.
Pendant le processus de simulation, nous comparons le voisinage du pixel simulé avec les représentants des *k* classes et trouvons le plus proche. Finalement on ne compare voisinage du pixel visité qu'avec le groupe du représentant le plus proche.
La recherche du meilleur motif est limitée à ce groupe, conduisant ainsi à une accélération significative de la simulation.

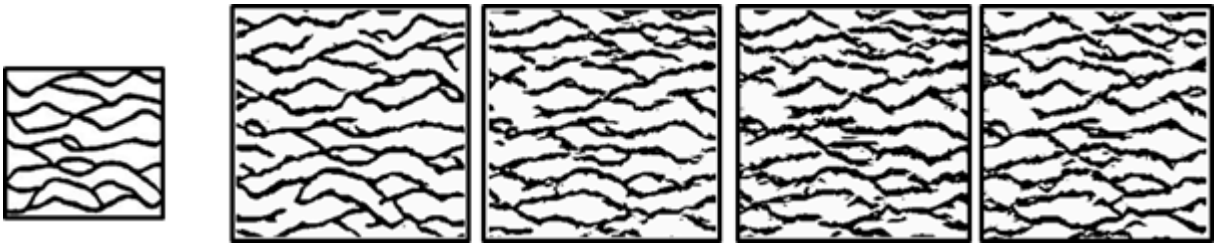

Figure 57 Impact du nombre de classes sur l'image synthétisée. a) l'image d'entrainement b) une classe c) 10 classes d) 20 classes e) 30 classes. La simulation à l'échelle grossière utilise un template 9 × 9 et un patch 3 × 3. La réalisation fine utilise un double template de 9 × 9 et 3 × 3, avec un patch 3 × 3.

Table 8 Comparaison du temps de calcul suivant le nombre de classes utilisées pour trier les dual-templates

Nombre de classes	1 classe	10 classes	20 classes	30 classes
Temps	T_2	0.43 T_2	0.32 T_2	0.19 T_2

Dans la Figure 57, nous montrons l'impact de la classification sur la qualité de la réalisation fine finale. Toutes les réalisations sont simulées avec le même ensemble de paramètres. Le seul changement est le nombre de classes dans le k-means. De toute évidence la réduction du nombre de motifs lors de la comparaison réduit la chance d'avoir le meilleur motif possible et devrait comme dans la section 4.4.1 dégrader la qualité de l'image fine finale. Toutefois nous pouvons voir que même avec 30 classes les chenaux sont encore continus et toujours connectés. Il y a plus d'interruptions brusques de chenaux, et plus de pixels isolés (bruits), et la réalisation finale est encore très similaire à l'image d'entrainement. Dans la Table 8, on rapporte les temps relatifs. La simulation avec 30 classes permet de diviser le temps CPU par 5 en comparaison avec la simulation sans classification.

5.6. Conclusion

Dans ce chapitre on a présenté un algorithme multipoint multi-échelles. Pour simplifier nous ne l'avons illustré que sur deux échelles, une fine correspondant à la résolution de l'image d'entrainement originale, et une grossière de résolution inférieure calculée à partir de l'image d'entrainement fine. L'algorithme se déroule alors en quatre étapes comme suit. Premièrement, on calcule une image d'entrainement grossière à partir de l'image d'entrainement fine. Deuxièmement, on simule la réalisation à l'échelle grossière en appliquant l'algorithme mono-

échelle. On utilise alors l'image d'apprentissage grossière. A partir de l'étape trois, on rentre dans le processus multi-échelles. Troisièmement, on construit la base de données multi-échelles à partir des motifs extraits des deux images d'entrainement. Et quatrièmement on simule la réalisation fine, à partir des images d'entrainement fine et grossière, conditionnée par la réalisation grossière que l'on vient de simuler. Des tests ont été réalisés pour mettre en évidence les avantages d'une approche multi-échelles en comparaison d'une approche mono-échelle. L'avantage principal indéniable est que l'algorithme multi-échelles capture les grandes hétérogénéités avec de plus petits templates, ce qui diminue considérablement les temps de calcul.

Un des points clé de la méthode multi-échelles, est l'obtention de l'image d'entrainement à l'échelle grossière, que ce soit pour des propriétés continues ou discrètes. Premièrement, on a étudié une image satellite d'une partie du delta du Gange particulièrement mal contrastée, pour mettre en évidence un problème qui peut arriver avec les variables continues. L'application d'un filtre avant la prise de moyenne, pour augmenter les contrastes et faire ressortir les structures principales de l'image, améliore la reproduction de ces grandes structures à l'échelle grossière et donc à l'échelle fine. Deuxièmement, on a étudié une image d'un réseau de fractures. On a vu que c'est la prise de médiane à l'échelle grossière qui permet d'obtenir une réalisation fine la plus semblable au réseau de l'image d'entrainement fine.

Pour terminer nous avons testé les techniques d'accélération décrites dans le chapitre précédent (Chapitre 4). Elles sont toutes deux basées sur la même idée de réduction du nombre de motifs à comparer lors de la simulation d'un patch. Dans la première approche, on ne scanne qu'une partie de la base de données, sélectionnée aléatoirement. Dans la seconde approche, on réalise d'abord une classification par classes de la base de données, et on ne scanne qu'une classe lors de la simulation du patch. Ces deux méthodes permettent de réduire significativement les temps de calcul. Cependant contrairement à la méthode mono-échelle, les réalisations finales sont beaucoup moins dégradées. En effet la réalisation à l'échelle grossière étant simulée très rapidement, il n'y a pas besoin de la modifier, elle reste donc de bonne qualité. Le point clé de l'approche multi-échelle est de simuler la réalisation fine avec comme contrainte la réalisation grossière. Si la réalisation à l'échelle grossière est de bonne qualité, elle conditionne très fortement la réalisation à l'échelle fine qui est alors de bonne qualité.

Conclusion Générale - Perspectives

La modélisation de réservoir est une étape importante dans l'exploration et l'exploitation d'un gisement d'hydrocarbures. C'est l'image du réservoir que l'on va construire à partir de toutes les données disponibles, et qui va servir à guider la gestion du champ. Vouloir obtenir l'image la plus réaliste et la plus exacte possible, le plus rapidement possible est alors compréhensible.

Nous avons proposé deux méthodes de simulation multi-échelles de réservoir. L'idée basée sur l'observation de l'architecture complexe des réservoirs et des différentes résolutions des données disponibles, était de copier dans un modèle cette structure à plusieurs échelles dépendantes les unes des autres, et bien évidemment d'en tirer partie en prenant en compte les grandes hétérogénéités aux échelles de plus basses résolutions pour diminuer les nombre de paramètres à caler.

Tout d'abord nous avons décrit la méthode de simulation séquentielle Gaussienne. C'est une technique qui s'inscrit dans la famille des méthodes géostatistiques à deux points. L'algorithme multi-échelles proposé permet de simuler une variable à l'échelle fine à partir d'une variable moyenne simulée au préalable à l'échelle grossière. Le point clé de ce passage d'une échelle à l'autre, est le calcul de la covariance croisée et de la covariance à l'échelle grossière, comme nous l'avons montré. L'un des avantages de cet algorithme est qu'il permet de modifier les réalisations au niveau des deux échelles simultanément ou séparément. La mise en place d'un cas synthétique et d'une boucle de calage d'historique a montré que la possibilité d'apporter des modifications sur les réalisations à l'échelle grossière s'avère particulièrement avantageuse. Cependant sur ce cas synthétique la comparaison entre le calage de l'échelle grossière en multi-échelle et le calage de l'échelle fine en mono-échelle, ne permet pas de conclure sur un quelconque avantage du calage multi-échelle sur le calage mono-échelle. Cette étude a soulevé aussi de nombreuses questions qu'il reste à étudier, telles le passage à une grille 3D, l'étude d'un cas plus réaliste.

Ensuite nous avons testé une adaptation de la synthèse de texture aux problématiques du réservoir, avec l'algorithme de simulation multipoint par patch. Cette méthode est fondée sur l'utilisation d'une image d'entrainement pour construire des images de structures complexes comme les méandres, les chenaux, les fractures, *etc*. L'image d'entrainement représente les structures que l'on attend dans le réservoir. Nous avons tout d'abord testé la sensibilité de l'algorithme aux différents paramètres, avant de proposer une nouvelle méthode d'intégration des données qui combine les avantages du chemin régulier avec l'obligation de trouver un motif qui s'ajuste aux données dures. Nous avons aussi combiné cet algorithme avec la méthode de déformation graduelle, pour se donner la possibilité de l'inclure dans un calage d'historique de production par exemple. Enfin nous avons implémenté deux techniques d'accélération de l'algorithme, qui sont le scan partiel de l'image d'entrainement et la classification par k-means de la base de données. Ces techniques permettent effectivement de diviser jusqu'à 5 fois le temps de calcul mais ne préservent pas la qualité de l'image finale. Le compromis pour garder

une image finale ressemblante est de scanner une assez grande partie de l'image ou d'introduire le concept de cohérence dans la classification. Cet algorithme montre des résultats assez bons mais des temps de calcul encore trop importants, et ne gère pas au mieux certaines images contenant de grandes structures.

Finalement, nous proposons une version multi-échelles de l'algorithme précédent, réduite à deux échelles par souci de simplification. Il faut deux images d'entrainement de résolution celle des échelles, *ie* il faut une image d'entrainement de résolution élevée pour la réalisation à l'échelle fine et une image d'entrainement de résolution plus basse pour la réalisation à l'échelle grossière. Dans une première étape on simule la réalisation grossière à partir de l'image d'entrainement grossière avec l'algorithme mono-échelle. Cette simulation est rapide, car l'image d'entrainement et la grille de simulation sont plus petites. Surtout elle permet de capturer les grandes hétérogénéités avec un template plus petit. Ensuite, il faut créer une base de données un peu plus complexe. Elle doit lister les motifs extraits de l'image d'entrainement fine, et les faire correspondre aux motifs extraits de l'image d'entrainement grossière, pour avoir des motifs double (fin et grossier) qui représente la même chose mais à deux résolutions différentes. La dernière étape est la simulation de l'échelle fine à partir des deux images d'entrainement, conditionnée par la réalisation à l'échelle grossière que l'on vient de simuler. Cette étape est plus rapide que dans le cas mono-échelle car le double template utilisé est plus petit que le template mono-échelle. En effet les grandes structures ont été capturées à l'échelle grossière, il suffit donc de rajouter les détails.

L'étape clé de cette méthode est donc de capturer les grandes lignes de l'architecture de l'image pendant la simulation à l'échelle grossière. Nous avons donc testé différentes manières d'obtenir une image d'entrainement grossière à partir de l'image d'entrainement fine, sur des variables continues et sur des variables discrètes, et leur impact sur la réalisation finale. Pour des variables continues on a montré que dans le cas où l'image était peu contrastée on pouvait rencontrer quelques problèmes pour simuler de bonnes images. Une des solutions proposée est de filtrer l'image pour extraire les grandes structures à l'échelle grossière. Pour des variables binaires, ici des fractures, nous avons obtenu des résultas contraires à notre intuition première. En effet, l'image d'apprentissage grossière obtenue par la médiane, ne ressemble pas du tout à l'image d'apprentissage fine et pourtant c'est elle qui donne les meilleurs résultats.

Pour finir, même si cette méthode est plus rapide que l'approche mono-échelle, on a testé les deux techniques d'accélération précédemment utilisées. Ces techniques ont démontré qu'elles permettaient de diviser le temps de calcul par 5 mais que dans le cas mono-échelle qu'elles dégradaient beaucoup trop l'image finale. Ici, la réalisation fine étant contrainte par la réalisation grossière, l'impact de la réduction du nombre de motifs scannés est moindre. Même avec seulement un quart de l'image scannée, on obtient une réalisation finale tout à fait acceptable.

Perspectives

L'objectif principal de cette approche multi-échelles est bien évidemment d'améliorer le processus du calage. Nous espérons que l'introduction d'une échelle grossière donnera plus de flexibilité au calage, le rendra plus efficace car il y aura moins de paramètres à modifier. Nous ne pourrons vraiment commencer à répondre à cette question que lorsque les méthodes décrites ci-dessus seront consolidées et appliquées à des cas tests plus réalistes. Et il reste encore à travailler l'intégration des données dures et molles dans le cas multi-échelles, ainsi que le passage à la 3D.

Nous avons vu le développement de modèles hiérarchiques qui mixaient les différentes techniques de simulation. Il serait intéressant de voir les résultats obtenus en simulant l'étape à l'échelle grossière avec par exemple de la synthèse de texture et l'étape à l'échelle fine avec du SGSim. Rappelons que SGSim apporte plus de variabilité dans les résultats que la synthèse de texture (point qui lui est souvent reproché).
A partir d'une image binaire de chenaux, on simulerait à l'échelle grossière l'emplacement des chenaux par synthèse de texture. Cette simulation est très rapide puisque l'image d'entrainement a une faible résolution. Ensuite la deuxième étape demande quelques tests mais l'idée serait d'utiliser l'information à l'échelle grossière pour contraindre la réalisation à l'échelle fine en simulant avec SGSim multi-échelles.

Références

Aanonsen, S.I., Eydinov, D., 2006, A multiscale method for distributed parameter estimation with application to reservoir history matching, Comput. Geosci., 10, 97-117.

Aradhye, H.B, Bakshi, B.R, Strauss, R.A, Davis, J.F., 2003, Multiscale SPC using wavelets: theoretical analysis and properties. Process system engineering, 49(4), 939-958

Arpat, G.B., Caers, J., 2005, A multiple-scale, pattern-based approach to sequential simulation, Geostatistics Banff 2004, Quantitative Geology and Geostatistics, 14(1), 255-264

Arpat, G.B., Caers, J., 2007, Conditional simulation with patterns, Mathematical Geology, 39(2), 177-203

Ashikhmin, M., 2001, Synthesizing Natural Textures, The proceedings of 2001 ACM Symposium on Interactive 3D Graphics, Research Triangle Park, NorthCarolina March 19-21, 217-226

Bar-Joseph, Z., El-Yaniv, R., Lischinski, D., Werman, M., 2001, Texture mixing and texture movie synthesis using statistical learning, IEEE Transactions on Visualization and Computer Graphics, 7, 2, 120-135

Behrens, R., Tran, T., 1998, Incorporating seismic data of intermediate vertical resolution into 3D reservoir models. In proceedings of the SPE 49143: Society of Petroleum Engineers Annual Technical conference and Exhibition, New Orleans, LA, USA

Bissell, R.C., Sharma, Y., Killough, J.E., 1994, History Matching using method of gradients: two cases studies, SPE Annual Technical Conference and Exhibition.

Blinn, J.F., Newell, M.E., 1976, Texture and reflection in computer generated images, Communication of the ACM 19, 10 (December), 542-547.

Bouzarkouna, Z., 2012, Well placement optimization, PhD thesis, Université Paris Sud 11

Bridge, J.S., Leeder, M.R., 1979, A simulation model of alluvial stratigraphy: Sedimentology, 26, 617-644

Buades, A., 2005, a non-local algorithm for image denoising, Computer Vision and Pattern Recognition, 2, 60-65, doi: 10.1109/CVPR.2005.38

Burt, P.J., Adelson, E.H., 1983, The Laplacian pyramid as a compact image code, IEEE Transaction on Communications, 31, 4, 532-540

Caers, J., 2003, Geostatistical history matching under training-image based geological constraints, SPE J., 8(3), 218-226

Carrera, J., Neuman, S.P., 1986, Estimation of aquifer parameters under transient and steady state conditions, Water Resources Research, 22(2).

Catmull, E.E., 1974, A Subdivision Algorithm for Computer Display of Curved Surfaces, PhD Thesis, Computer Science Department, University of Utah, Salt Lake City.

Cellis, M.R., Dennis, J.E., Tapia, R.A., 1985, A trust region strategy for non-linear equality constrained optimization, Numerical Optimization, 71-82

Chatterjee, S., Dimitrakopoulos, R., Mustapha, H., 2012, Dimensional reduction of pattern-based simulation using wavelet analysis, Mathematical Geosciences, 44, 343-374

Conn, A., Scheinberg, K., Vicente, L., 2009, Introduction to derivative free optimization, MPS-SIAM Series on optimization.

Corey, A.T., 1977, Mechanics of heterogeneous fluids in porous media, Water Resources publications, Fort Collins, Colorado.

Crouse, M.S., Nowak, R.D., Baraniuk, R.G., 1998, Wavelet-based statistical signal processing using hidden Markov models, IEEE Trans Signal Process, 46(4), 886-902

De Bonet, J., 1997, Multiresolution Sampling procedure for analysis and synthesis of texture images, In SIGGRAPH97, pp361-368

De Marsily, G.,Delay, F., Gonçalvès,J., Renard, P., Teles, V., Violette, S., 2005, Dealing with spatial heterogeneity, Hydrogeology Journal, 13(1), 161-183

De Montleau, P., Cominelli, A., Neylon, D.R.K., Pallester, I., Tesaker, O., Nygoud, I., 2006, Production optimization under constraints using adjoint gradients, Proceedings of 10th ECMOR

Desbarats, A.J., 1996, Modeling spatial variability using geostatistical simulation, In: Rouahni, S., Srivastava, R.M., Desbarats, A.J., Cromer, M.V., Johnson, A.I., Geostatistic for Environmental and Geotechnical Applications, American Society for Testing and Materials STP 1283, Philadelphia, pp32-48.

Deutsch, C.V., Lewis, R., 1992, Advances in the practical implementation of indicator geostatistics, In: proceedings of the 23rd International APCOM Symposium, Tucson, AZ, Society of Mining Engineers, pp 169-179

Deutsch, C.V., Wang, L., 1996, Hierarchical Object-Based Stochastic Modeling of Fluvial Reservoirs, Mathematical Geology, 28(7), 857-880

Deutsch, C. V., and Journel, A.G., 1998, GSLIB: Geostatistical software library and user's guide, 2nd ed.: Oxford University Press, New York, 369 p.

Droujinine, A., 2006, Multiscale geophysical data analysis using the Eigen image discrete wavelet transform. Journal of Geophysics and Engineering, 3, 59-69

Duda, R.O., Hart, P.E., Stork, D.G., 2001, Pattern classification, edited by Wiley-interscience

Duranleau, F., 2008, Génération et édition de textures géométriques représentées par des ensembles de points, Thèse de doctorat, Département d'informatique et de recherche opérationnelle, Université de Montréal.

Efendiev,Y., Hou, T.Y., Luo, W., 2006, Preconditioning Markov chain Monte Carlo simulations using coarse-scale models, SIAM J. Sci. Comp., 28, 776-803

Efros, A.A., Leung, T.K., 1999, Texture synthesis by non-parametric sampling, IEEE International Conference on Computer Vision, Corfu, Greece

Efros, A.A., Freeman, W.T., 2001, Image quilting for texture synthesis and transfer, paper presented at Proceedings of ACM SIGGRAPH Conference on computer graphics.

El Ouassini, A., Saucier, A., Marcotte, D., Favis, B., 2008, A patchwork approach to stochastic simulation: A route towards the analysis of morphology in multiphase systems, Chaos Solitons and Fractal, 36(2), 418-436

Evensen, G. 1994. Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics, J. Geophys. Res., 99, 10,143-10,162

Evensen, G., 2009, Data assimilation, the Ensemble Kalman Filter, 2nd Edition, Springer.

Fenwick, D., Thiele, M., Agil, M., Hussain, A.P., Humam, F., Caers, J., 2005, Reconciling prior geologic information with production data using streamlines: application to a giant Middle-Eastern oil field, SPE ATCE, Dallas, TX, USA, SPE 95940.

Froidevaux, R., 1992, Probability field simulation, Geostatistics Troia'92, proceedings of the Fourth Geostatistics Congress, A. Soares (ed.), Kluwer Academic Publishers, Dordrecht, Netherland.

Gao, G., Li, G., Reynolds, A.C., 2007, A stochastic optimization algorithm for automatic history matching, SPE Journal, 12(2), 196-208

Garcia, M., Froidevaux, R., 1997, Application of geostatistics to 3D modelling of contaminated sites: a case study, In: Soares, A., Gomez-Hernandez, J., Froidevaux, R., geoENV I – Geostatistics for Environmental Applications, Kluwer Academic publishers, Dordrecht, pp309-325.

Gardet, C., Le Ravalec, M., 2014, Multiscale Multiple Point Simulation Based on Texture Synthesis, 14th European Conference on the Mathematics of Oil Recovery Catania, Sicily, Italy, 8-11 September

Gardet, C., Le Ravalec, M., Gloaguen, E., 2014, Multiscale Parameterization of Petrophysical Properties for Efficient History-Matching, Mathematical Geosciences, 46(3), 315-336, doi: 10.1007/s11004-013-9480-3

Gervais, V., Roggero, F., 2010, Integration of saturation data in an history matching process based on adaptive local parameterization, J. Pet. Sci. Eng., 73(1-2), 86-98.