MODÉLISATION ET INVERSION EN GÉOPHYSIQUE
6 - Inversion non linéaire

Bernard Giroux
(bernard.giroux@ete.inrs.ca)

Institut national de la recherche scientifique
Centre Eau Terre Environnement

Version 1.0.1
Hiver 2018

Régression non linéaire

Aperçu

- Les problèmes non linéaires n'obéissent pas aux principes de superposition et de mise à l'échelle ;
- Le modèle direct $G(\mathbf{m})$ *n'est plus* un système linéaire d'équations algébriques $\mathbf{Gm} = \mathbf{d}$.
- Il n'existe pas de théorie générale donnant une solution pour les problèmes inverses non linéaires ;
- Il existe néanmoins plusieurs approches, certaines basées sur des approximations linéaires, qui sont souvent applicables.

Méthode de Newton

- L'idée de la méthode de Newton est d'utiliser une information sur la forme de l'erreur de prédiction $E(\mathbf{m})$ au voisinage d'une solution d'essai $\mathbf{m}^{(p)}$ pour trouver une meilleure solution $\mathbf{m}^{(p+1)}$.
- Les dérivées de $E(\mathbf{m})$ nous renseignent sur sa forme.
- À partir de l'expansion en série de Taylor de $E(\mathbf{m})$ au voisinage de $\mathbf{m}^{(p)}$ et en gardant les trois 1^{er} termes, on a

$$E(\mathbf{m}) \approx E(\mathbf{m}^{(p)}) + \sum_{i=0}^{M-1} b_i \left(m_i - m_i^{(p)}\right) + \frac{1}{2} \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} H_{ij} \left(m_i - m_i^{(p)}\right) \left(m_j - m_j^{(p)}\right) \quad (1)$$

où $b_i = \left.\frac{\partial E}{\partial m_i}\right|_{\mathbf{m}^{(p)}}$ et $H_{ij} = \left.\frac{\partial^2 E}{\partial m_i \partial m_j}\right|_{\mathbf{m}^{(p)}}$.

Méthode de Newton

- Sous forme matricielle, l'équation précédente s'écrit

$$E(\mathbf{m}^{(p)}) + \nabla E(\mathbf{m}^{(p)})^T \Delta\mathbf{m} + \frac{1}{2}\Delta\mathbf{m}^T \mathbf{H}\left(E(\mathbf{m}^{(p)})\right)\Delta\mathbf{m} \tag{2}$$

où

$$\Delta\mathbf{m} = \mathbf{m} - \mathbf{m}^{(p)} \tag{3}$$

et $\nabla E \equiv \nabla E(\mathbf{m}^{(p)})$ est le gradient, et $\mathbf{H} \equiv \mathbf{H}\left(E(\mathbf{m}^{(p)})\right)$ est la hessienne, avec :

$$\nabla E = \left[\begin{array}{c} \frac{\partial E}{\partial m_0} \\ \vdots \\ \frac{\partial E}{\partial m_{M-1}} \end{array}\right]\Bigg|_{\mathbf{m}^{(p)}} \qquad \mathbf{H} = \left[\begin{array}{ccc} \frac{\partial^2 E}{\partial m_0^2} & \cdots & \frac{\partial^2 E}{\partial m_0 \partial m_{M-1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 E}{\partial m_{M-1}\partial m_0} & \cdots & \frac{\partial^2 E}{\partial m_{M-1}^2} \end{array}\right]\Bigg|_{\mathbf{m}^{(p)}} \tag{4}$$

Méthode de Newton

- Le minimum de $E(\mathbf{m})$ peut maintenant être trouvé en dérivant à nouveau et en égalant à zéro :

$$\frac{\partial E(\mathbf{m})}{\partial m_q} = 0 = b_q + \sum_{j=0}^{M-1} H_{qj} \left(m_j - m_j^{(p)} \right). \tag{5}$$

- Sous forme matricielle, cela donne

$$\Delta \mathbf{m} = -\mathbf{H}^{-1} \nabla E \tag{6}$$

- Notez que pour le cas linéaire $\mathbf{Gm} = \mathbf{d}$, on peut trouver que $\nabla E = -2\mathbf{G}^T(\mathbf{d} - \mathbf{Gm}^{(p)})$ et que $\mathbf{H} = 2\mathbf{G}^T\mathbf{G}$, ce qui nous amène à $\mathbf{m} = \left[\mathbf{G}^T\mathbf{G}\right]^{-1}\mathbf{G}^T\mathbf{d}$, soit la solution des moindres-carrés.

Méthode de Newton

- L'algorithme de Newton est le suivant

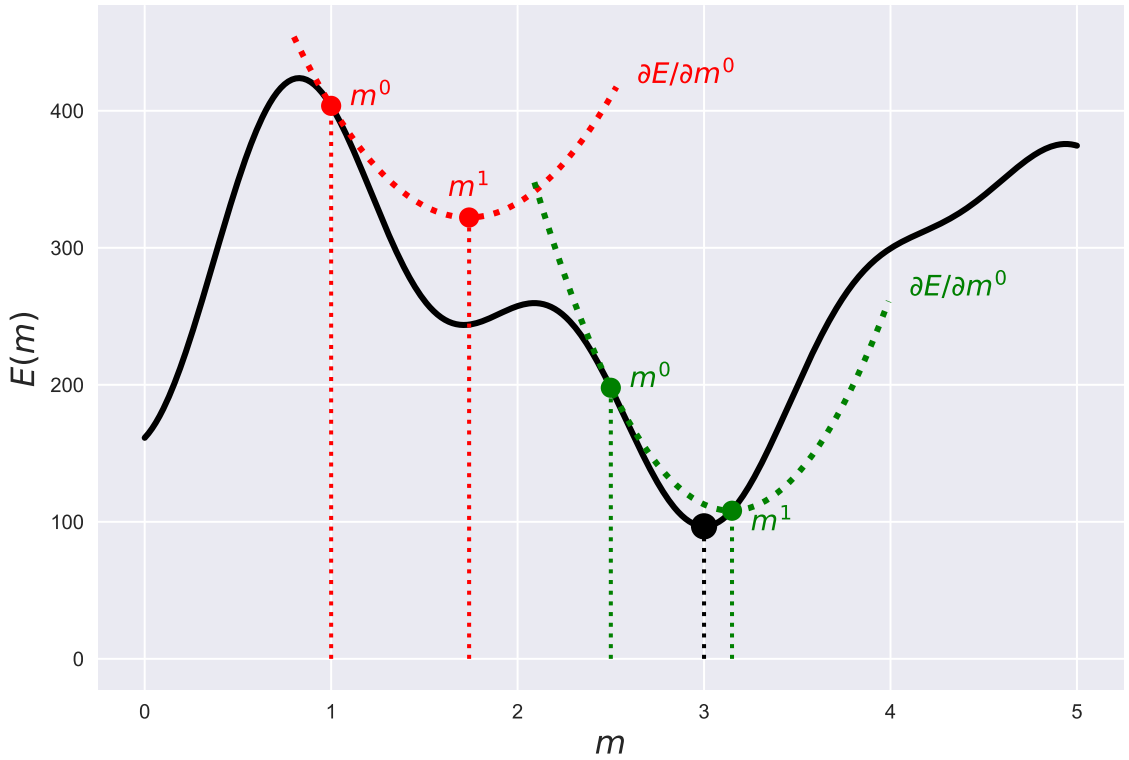
Algorithm 1 Méthode de Newton

1: $p \leftarrow 0$
2: **while** $\nabla E \neq 0$
3: Calculer ∇E et $\mathbf{H}$
4: Résoudre $\mathbf{H}\,\Delta\mathbf{m} = -\nabla E$
5: $\mathbf{m}^{(p+1)} \leftarrow \mathbf{m}^{(p)} + \Delta\mathbf{m}$
6: $p \leftarrow p + 1$

- Cet algorithme peut être très efficace, mais la convergence n'est pas garantie.
- L'algorithme peut aussi converger vers un minimum local si la solution initiale est trop loin du minimum global.

Méthode de Newton

- Exemple de cas où une solution initiale converge vers un minimum local (rouge), alors qu'une solution initiale plus proche du minimum global converge mieux (vert).



Méthode de Newton

- En pratique, la convergence $\nabla E = 0$ n'est jamais atteinte ;
- En général, on considère qu'il y a convergence lorsqu'une des conditions suivante est atteinte :
  - $\max |\nabla E| < \epsilon_1$
  - $\max |\Delta m_i / m_i| < \epsilon_2$
  - $E/(N - M + 1) < \epsilon_3$
  - $|E^{(p)} - E^{(p-1)}| < \epsilon_4 E^{(p)}$ et $E^{(p)} < E^{(p-1)}$

  sinon les itérations s'arrêtent après un nombre maximal prédéfini.

Méthode de Gauss-Newton

- La méthode de Newton n'est pas applicable quand il n'y a pas de solution exacte à $G(\mathbf{m}) = \mathbf{d}$ ou que plusieurs solutions existent.
- La méthode de Gauss-Newton peut être vue comme une modification permettant de minimiser les moindres-carrés non linéaires.
- On cherche à minimiser la norme des résidus pondérés

$$E = \sum_{i=0}^{N-1} \left(\frac{d_i - \hat{d}_i}{\sigma_i} \right)^2 , \tag{7}$$

où $\hat{d}_i = G_i(\mathbf{m})$ et σ_i est l'écart-type de la i^e mesure.

Méthode de Gauss-Newton

- On défini la fonction scalaire

$$E_i(\mathbf{m}) = \frac{d_i - \hat{d}_i}{\sigma_i} \quad i = 0, 1, \ldots, N-1 \tag{8}$$

et la fonction vecteur

$$\mathbf{E}(\mathbf{m}) = \begin{bmatrix} E_0(\mathbf{m}) \\ \vdots \\ E_{N-1}(\mathbf{m}) \end{bmatrix} \tag{9}$$

- Ainsi

$$E = \sum_{i=0}^{N-1} E_i(\mathbf{m})^2 = \|\mathbf{E}(\mathbf{m})\|_2^2 = (\mathbf{d} - \hat{\mathbf{d}})^T \mathbf{W} (\mathbf{d} - \hat{\mathbf{d}}) \tag{10}$$

où $\mathbf{W}$ est une matrice diagonale avec $W_{ii} = 1/\sigma_i^2$.

Méthode de Gauss-Newton

- Le gradient de E est la somme des gradients des fonctions :

$$\nabla E = \sum_{i=0}^{N-1} \nabla \left(E_i(\mathbf{m})^2 \right) \tag{11}$$

et les éléments du gradient sont

$$\left(\nabla E(\mathbf{m})\right)_j = \sum_{i=0}^{N-1} 2E_i(\mathbf{m}) \left(\nabla E_i(\mathbf{m})\right)_j \tag{12}$$

où l'indice j signifie $\frac{\partial}{\partial m_j}$ et où $j = 0, 1, \ldots, M-1$.

Méthode de Gauss-Newton

- Sous forme matricielle, on trouve

$$\nabla E = 2\mathbf{W}^{1/2}\mathbf{J}^T\mathbf{E}(\mathbf{m}) = 2\mathbf{J}^T\mathbf{W}(\mathbf{d} - \hat{\mathbf{d}}) \tag{13}$$

où $\mathbf{J}$ est la matrice jacobienne $(N \times M)$:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial \hat{d}_0}{\partial m_0} & \cdots & \frac{\partial \hat{d}_0}{\partial m_{M-1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \hat{d}_{N-1}}{\partial m_0} & \cdots & \frac{\partial \hat{d}_{N-1}}{\partial m_{M-1}} \end{bmatrix} \tag{14}$$

Méthode de Gauss-Newton

- D'une façon similaire, on peut exprimer la hessienne de $E(\mathbf{m})$ à partir de ses fonctions :

$$\mathbf{H}\left(E(\mathbf{m})\right) = \sum_{i=0}^{N-1} \mathbf{H}\left(E_i(\mathbf{m})^2\right) \tag{15}$$

$$= \sum_{i=0}^{N-1} \mathbf{H}^i(\mathbf{m}) \tag{16}$$

où $\mathbf{H}^i(\mathbf{m})$ est la hessienne de $E_i(\mathbf{m})^2$.

Méthode de Gauss-Newton

- Les éléments j,k de $\mathbf{H}^i(\mathbf{m})$ sont

$$\mathbf{H}^i_{j,k}(\mathbf{m}) = \frac{\partial^2 \left(E_i(\mathbf{m})^2\right)}{\partial m_j \partial m_k} \tag{17}$$

$$= \frac{\partial}{\partial m_j}\left(2E_i(\mathbf{m})\frac{\partial E_i(\mathbf{m})}{\partial m_k}\right) \tag{18}$$

$$= 2\left(\frac{\partial E_i(\mathbf{m})}{\partial m_j}\frac{\partial E_i(\mathbf{m})}{\partial m_k} + E_i(\mathbf{m})\frac{\partial^2 E_i(\mathbf{m})}{\partial m_j \partial m_k}\right) \tag{19}$$

- Sous forme matricielle, cela devient

$$\mathbf{H}\left(E(\mathbf{m})\right) \equiv \mathbf{H} = 2\mathbf{J}^T\mathbf{W}\mathbf{J} + \mathbf{Q}(\mathbf{m}) \tag{20}$$

où

$$\mathbf{Q}(\mathbf{m}) = 2\sum_{i=0}^{N-1} E_i(\mathbf{m})\mathbf{H}\left(E_i(\mathbf{m})\right). \tag{21}$$

Méthode de Gauss-Newton

- Avec la méthode de Gauss-Newton, on ignore la matrice $\mathbf{Q}(\mathbf{m})$
  - Cela revient à assumer que la fonction est quasi linéaire *au voisinage* de $\mathbf{m}^{(p)}$.
- La matrice hessienne devient

$$\mathbf{H} \approx 2\mathbf{J}^T\mathbf{W}\mathbf{J} \tag{22}$$

- La mise à jour du modèle est obtenue en résolvant

$$\mathbf{J}^T\mathbf{W}\mathbf{J}\Delta\mathbf{m} = \mathbf{J}^T\mathbf{W}(\mathbf{d} - \hat{\mathbf{d}}) \tag{23}$$

- Comme pour la méthode de Newton, la méthode de Gauss-Newton peut converger vers un minimum local.

Méthode de Gauss-Newton – Exercice

- Soit une fonction non linéaire

$$G_i(\mathbf{m}) = \sin(\omega_0 m_0 x_i) + m_0 m_1 \tag{24}$$

- Quel est le minimum de E après 20 itérations pour les données bruitées suivantes, en utilisant $\mathbf{m}^{(0)} = [1.0, 1.0]$ et en assumant que $G(\mathbf{m})$ est quasi-linéaire et que $\mathbf{W} = \mathbf{I}$:

```
N = 40
xmin = 0
xmax = 1.0
dx = (xmax-xmin)/(N-1)
x = dx*np.arange(N)

mt = [1.21, 1.54] # modèle vrai

w0 = 20
dtrue = np.sin(w0*mt[0]*x) + mt[0]*mt[1]   # données propres
sd = 0.4
dobs = dtrue + sd*np.random.randn(N)   # données bruitées
```

Calcul de la jacobienne

- La méthode de Gauss-Newton requière le calcul de dérivées partielles pour construire la matrice jacobienne.
- Dans certains cas, les expressions analytiques de ces dérivées existent et il est possible de les calculer directement.
- Lorsque les expressions analytiques ne sont pas disponibles, on peut les calculer par différences finies :

$$\frac{\partial \hat{d}_i}{\partial m_j} = \frac{\partial G_i(\mathbf{m})}{\partial m_j} \approx \frac{\partial G_i(\mathbf{m} + \Delta \mathbf{m}_j) - G_i(\mathbf{m})}{\Delta m_j} \tag{25}$$

où $\Delta \mathbf{m}_j$ est le vecteur $\mathbf{m}$ perturbé de Δm_j uniquement à j.

Calcul de la jacobienne

- Le choix de la perturbation Δm_j peut s'avérer délicat lorsque les paramètres varient sur plusieurs ordres de grandeur ;
  - dans un tel cas, il est préférable que la perturbation soit une fraction de la valeur du paramètre plutôt qu'une valeur absolue.
- Lorsque les paramètres du modèle représentent des propriétés physiques différentes variant sur des ordres de grandeur différents, il peut être nécessaire de normaliser les termes de la jacobienne pour éviter de donner des poids trop différents.
- Il est également important que l'estimation des dérivées soit stable au voisinage de la perturbation.

Méthode de Levenberg-Marquardt

- Il peut arriver que la matrice $\mathbf{J}^T\mathbf{J}$ soit singulière ;
  - dans cette situation, la méthode de Gauss-Newton échoue.
- Avec la méthode de Levenberg-Marquardt, la mise à jour du modèle est

$$\left(\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I}\right)\Delta\mathbf{m} = \nabla E \tag{26}$$

  - Le terme $\lambda\mathbf{I}$ permet de mieux conditionner le système.
- Avec cette méthode, le problème du choix de la valeur optimale de λ apparaît ;
  - si λ est trop élevé, la convergence est ralentie ;
  - si λ est trop faible, le problème de la singularité peut survenir à nouveau.

Méthode de Levenberg-Marquardt

- Pour des problèmes de taille modérée et lorsque $\mathbf{W} = \mathbf{I}$, une façon de déterminer λ passe par la SVD ;
- La SVD de $\mathbf{J}$ est

$$\mathbf{J} = \mathbf{U}\mathbf{S}\mathbf{V}^T \tag{27}$$

où $\mathbf{U}$ contient les vecteurs de base de l'espace des données, $\mathbf{V}$ contient les vecteurs de base de l'espace des paramètres, et $\mathbf{S}$ contient les valeurs singulières $[s_0, s_1, \ldots, s_{M-1}]$ de $\mathbf{J}$.

- En insérant cette expression dans l'équation (26), on trouve

$$\Delta\mathbf{m} = \left(\mathbf{V}\mathbf{S}^2\mathbf{V}^T + \beta^2\mathbf{I}\right)^{-1}\mathbf{V}\mathbf{S}\mathbf{U}^T\left(\mathbf{d} - \hat{\mathbf{d}}\right) \tag{28}$$

où $\beta^2 = \lambda$.

Méthode de Levenberg-Marquardt

- En ajoutant le facteur β^2 sur la diagonale de $\mathbf{S}^2$, on trouve

$$\left(\mathbf{V}\mathbf{S}^2\mathbf{V}^T + \beta^2\mathbf{I}\right) = \left(\mathbf{V}\text{diag}(\mathbf{s}^2)\mathbf{V}^T + \beta^2\mathbf{I}\right) \tag{29}$$

$$= \mathbf{V}\text{diag}(\mathbf{s}^2 + \beta^2)\mathbf{V}^T \tag{30}$$

où $\mathbf{s}$ est le vecteur des valeurs singulières.

- L'inverse de cette expression est

$$\left(\mathbf{V}\text{diag}(\mathbf{s}^2 + \beta^2)\mathbf{V}^T\right)^{-1} = \mathbf{V}\text{diag}\left(\frac{1}{\mathbf{s}^2 + \beta^2}\right)\mathbf{V}^T \tag{31}$$

- En insérant cette dernière expression dans (28), on trouve

$$\Delta\mathbf{m} = \mathbf{V}\text{diag}\left(\frac{1}{\mathbf{s}^2 + \beta^2}\right)\mathbf{V}^T\,\mathbf{V}\mathbf{S}\mathbf{U}^T\left(\mathbf{d} - \hat{\mathbf{d}}\right) \tag{32}$$

$$= \mathbf{V}\text{diag}\left(\frac{\mathbf{s}}{\mathbf{s}^2 + \beta^2}\right)\mathbf{U}^T\left(\mathbf{d} - \hat{\mathbf{d}}\right) \tag{33}$$

Méthode de Levenberg-Marquardt

- L'estimation de β est dynamique et se fait en fonction de la convergence ;
- Une façon de déterminer β est d'utiliser

$$\beta = s_l \Delta E^{1/l} \tag{34}$$

où $l = 1, \ldots, M$ et

$$\Delta E^{(p)} = \frac{E^{(p-1)} - E^{(p)}}{E^{(p-1)}}. \tag{35}$$

Méthode de Levenberg-Marquardt

- La procédure est la suivante :

1: $\mathbf{m} \leftarrow \mathbf{m}^{(0)}$
2: **while** $p < p_{max}$
3: Calculer $\hat{\mathbf{d}}$ et $E^{(p)}$
4: **if** convergence atteinte
5: return
6: Calculer $\mathbf{J}$ et SVD($\mathbf{J}$)
7: $l \leftarrow 1, k \leftarrow 1$
8: **while** $l < M$
9: $\beta \leftarrow s_l \Delta E^{1/l}$
10: Calculer $\Delta\mathbf{m}$, $\hat{\mathbf{d}}$ et $E^{(p)}$
11: **if** $E^{(p)} > E^{(p-1)}$
12: $l \leftarrow l + 1, k \leftarrow k + 1$
13: **if** $k == M$
14: return
15: **else**
16: $\mathbf{m} \leftarrow \mathbf{m} + \Delta\mathbf{m}$
17: Calculer ΔE
18: $l \leftarrow M$
19: **if** convergence atteinte
20: return
21: $E^{(p-1)} \leftarrow E^{(p)}$
22: $p \leftarrow p + 1$

Méth. de Levenberg-Marquardt – Exercice

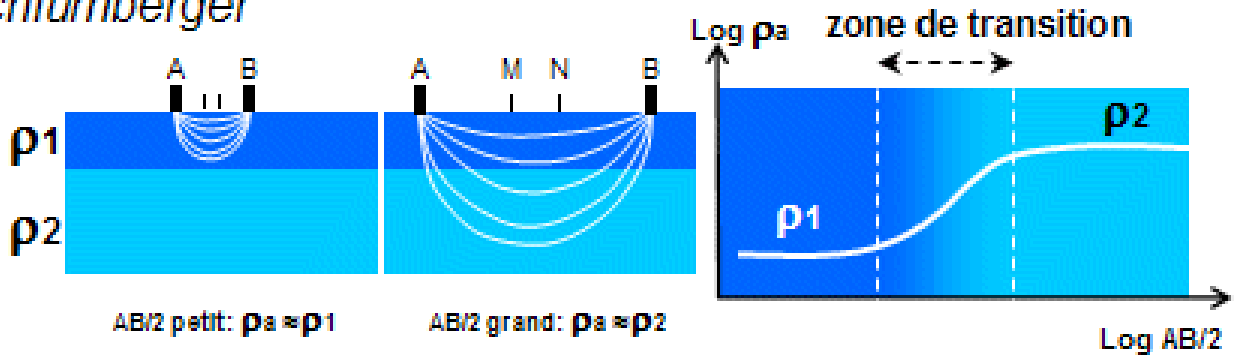
- Utilisez la méthode de Levenberg-Marquardt pour estimer un modèle géoélectrique 1D à partir de mesures de sondages Sclumberger.

résistivité apparente

$$\rho_a = \frac{\Delta V}{I}\left[\frac{2\cdot\pi}{\frac{1}{AM}-\frac{1}{AN}-\frac{1}{BM}+\frac{1}{BN}}\right]$$

Facteur K =géométrie du dispositif

Sondage électrique quadripôle Schlumberger



Source : Université de Lorraine

Méth. de Levenberg-Marquardt – Exercice

- Le fichier disponible à `https://github.com/bernard-giroux/geo1302/blob/master/ves_part.py` contient une classe `Sondage` avec les méthodes
  - `mod` pour modéliser la courbe de resistivité apparente ;
  - `_jacobian` pour calculer la jacobienne ;
  - `inv` pour faire l'inversion.
- La fonction `inv` est à compléter.
- Important :
  - la résistivité peut varier sur plusieurs ordres de grandeur ;
  - il est préférable de calculer l'erreur de prédiction avec le log de la résistivité (i.e. $\mathbf{d} = \log(\rho_a^{\text{obs}})$ et $\hat{\mathbf{d}} = \log(\rho_a^{\text{pre}})$) pour éviter de donner un poids relatif trop faible aux faibles résistivités ;
  - il faut alors en tenir compte lors de la mise à jour car $\Delta\mathbf{m}$ est obtenu à partir de $\mathbf{d} - \hat{\mathbf{d}}$

$$\mathbf{m} = \exp\left(\log(\mathbf{m}) + \Delta\mathbf{m}\right) \tag{36}$$

Inversion non linéaire

Régularisation

- Les méthodes vues précédemment fonctionnent bien lorsqu'il y a peu de paramètres à estimer.
- Les modèles comportant un plus grand nombre de paramètres sont susceptibles d'être partiellement indéterminés ;
  - comme pour le cas linéaire, il faut régulariser le problème.
- Les approches cherchant à minimiser la norme $\|\mathbf{Dm}\|_2$ (où $\mathbf{D}$ est une matrice de lissage) peuvent être adaptées au cas non linéaire.
- Le problème peut être formulé en posant

$$\min \|G(\mathbf{m}) - \mathbf{d}\|_2^2 + \alpha^2 \|\mathbf{Dm}\|_2^2 \tag{37}$$

e.g. sous une forme amortie.

Régularisation

- La méthode de Gauss-Newton peut être utilisée si on récrit (37) tel que

$$\min \left\| \begin{matrix} G(\mathbf{m}) - \mathbf{d} \\ \alpha \mathbf{D}\mathbf{m} \end{matrix} \right\|_2^2 \tag{38}$$

- La jacobienne de (38) à l'itération p est

$$\mathbf{K}^{(p)} = \begin{bmatrix} \mathbf{J}^{(p)} \\ \alpha \mathbf{D} \end{bmatrix} \tag{39}$$

- En insérant (39) dans (23), on obtient

$$\mathbf{K}^{(p)T}\mathbf{K}^{(p)}\Delta\mathbf{m} = -\mathbf{K}^{(p)T} \begin{bmatrix} G(\mathbf{m}^{(p)}) - \mathbf{d} \\ \alpha \mathbf{D}\mathbf{m}^{(p)} \end{bmatrix} \tag{40}$$

Régularisation

- En combinant (39) et (40), on trouve finalement

$$\left(\mathbf{J}^T\mathbf{J} + \alpha^2\mathbf{D}^T\mathbf{D}\right)\Delta\mathbf{m} = -\mathbf{J}^T\left(G(\mathbf{m}) - \mathbf{d}\right) - \alpha^2\mathbf{D}^T\mathbf{D}\mathbf{m} \quad (41)$$

  où l'indice (p) indiquant l'itération a été omis pour plus de clarté.
  - Il est important de noter que *la jacobienne doit être recalculée à chaque itération*.
- Le terme de régularisation α stabilise le système ;
  - en général il n'est pas nécessaire d'utiliser le terme $\lambda\mathbf{I}$ de Levenberg-Marquardt.

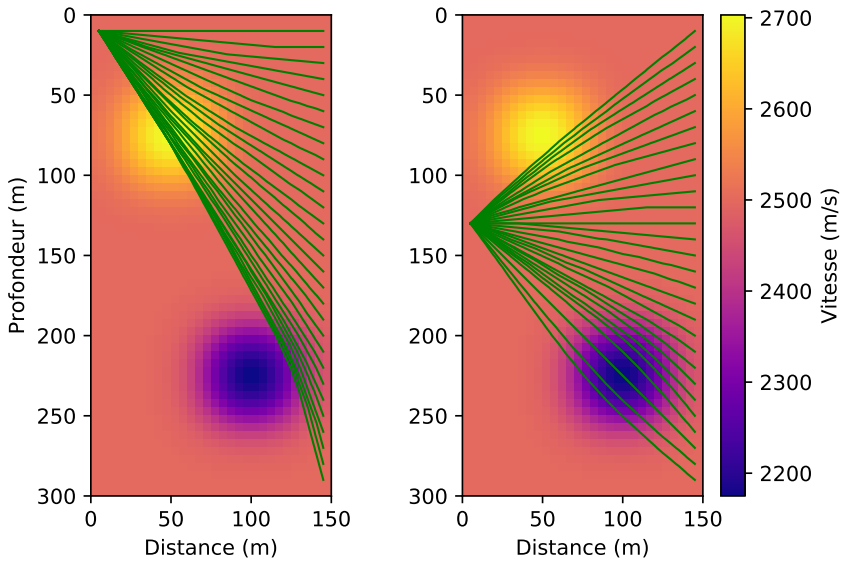
Régularisation

- Pour certaines applications, il peut être intéressant d'appliquer un lissage différent selon la direction ;
  - dans un milieu sédimentaire, un lissage horizontal plus marqué est souvent préférable car les paramètres varient moins horizontalement que verticalement.
- Cela se fait en définissant des matrices de lissage pour chaque direction et en utilisant un poids différent pour chacune d'elle :

$$\left(\mathbf{J}^T\mathbf{J} + \alpha^2\mathbf{D}_x^T\mathbf{D}_x + \beta^2\mathbf{D}_z^T\mathbf{D}_z\right)\Delta\mathbf{m} = -\mathbf{J}^T\left(G(\mathbf{m}) - \mathbf{d}\right) - \alpha^2\mathbf{D}_x^T\mathbf{D}_x\mathbf{m} - \beta^2\mathbf{D}_z^T\mathbf{D}_z\mathbf{m} \quad (42)$$

Régularisation – Exemple

- Considérons le cas de la tomographie entre forages pour lequel les rais sont infléchis en raison des contrastes de vitesse.
- On cherche à estimer la vitesse à partir des temps de parcours.



- Le temps de parcours de la source Tx au récepteur Rx est

$$t = \int_{Tx}^{Rx} s(l)\, dl \tag{43}$$

où s est la lenteur et l est la trajectoire.

Régularisation – Exemple

- Le modèle est discrétisé en cellules à l'intérieur desquelles la vitesse/lenteur est constante, i.e.

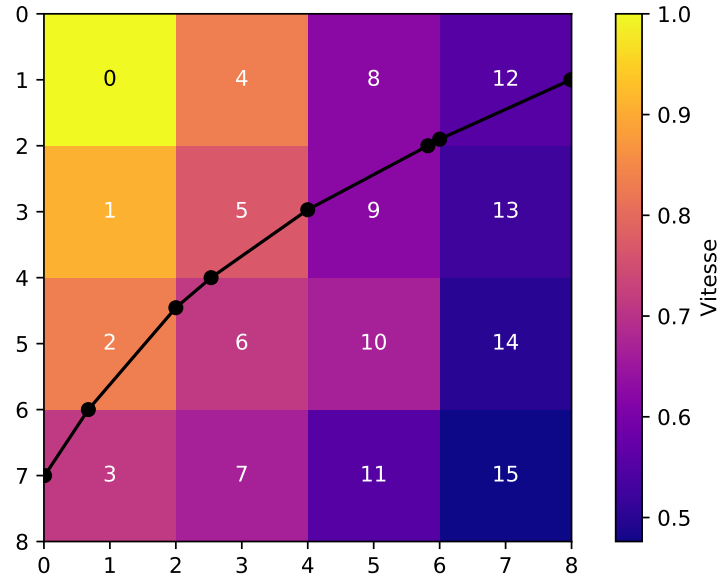
$$\mathbf{m} = [s_0, s_1, s_2, \ldots, s_{M-1}]^T \tag{44}$$

- Pour un tel modèle, le temps de parcours pour une paire $Tx - Rx$ devient

$$t = \sum_{i}^{nseg} s_i l_i \tag{45}$$

où i correspond aux cellules traversées par un segment de rai et l_i est la longueur du segment.

Régularisation – Exemple



- Pour le modèle ci-dessus, le temps est

$$\begin{aligned} t &= s_2 l_2 + s_3 l_3 + s_5 l_5 + s_6 l_6 + s_8 l_8 + s_9 l_9 + s_{12} l_{12} \\ &= 2.0 s_2 + 1.2 s_3 + 1.8 s_5 + 0.7 s_6 + 0.2 s_8 + 2.1 s_9 + 2.2 s_{12} \\ &= \mathbf{lm} \end{aligned}$$

où $\mathbf{l} = [0, 0, 2.0, 1.2, 0, 1.8, 0.7, 0, 0.2, 2.1, 0, 0, 2.2, 0, 0, 0]$

Régularisation – Exemple

- Pour plusieurs mesures à des paires $Tx - Rx$ différentes, on aura

$$\mathbf{d} = \mathbf{Lm} \tag{46}$$

où

$$\mathbf{d} = \left[t_0, t_1, t_2, \ldots, t_{N-1}\right]^T \tag{47}$$

et $\mathbf{L}$ est une matrice creuse $N \times M$ contenant les longueurs de segments.

- $\mathbf{L}$ doit être recalculée chaque fois que $\mathbf{m}$ est mis à jour.
- Il est intéressant de noter que la jacobienne est égale à $\mathbf{L}$, par exemple pour le 1^{er} terme on a

$$\frac{\partial d_0}{\partial m_0} = \frac{\partial}{\partial m_0} s_0 l_0 + s_1 l_1 + s_2 l_2 + \ldots = l_0 \tag{48}$$

Régularisation – Exemple

- Les données sont contenues dans le fichier `model1_tt.dat`

```
data = np.loadtxt('model1_tt.dat')
Tx = data[:, :2].copy()
Rx = data[:, 2:4].copy()
dobs = data[:, -1]
```

- Le modèle initial peut être défini à partir de la lenteur apparente moyenne :
  - la lenteur apparente est définie par le temps de parcours entre $Tx - Rx$ divisé par la distance en ligne droite entre $Tx - Rx$.
- La matrice $\mathbf{L}$ peut être obtenue à partir de la classe `Grid2Dcpp` du module `cgrid2d`

Régularisation – Exemple


```
# Paramètres de la grille
dx = 5.0
dz = 5.0
xmin = 0.0
zmin = 0.0
xmax = 150.0
zmax = 300.0
nx = int((xmax-xmin)/dx+0.001)
nz = int((zmax-xmin)/dz+0.001)
xg = np.linspace(xmin, xmax, nx+1)
zg = np.linspace(zmin, zmax, nz+1)

# Modèle initial
Ldroit = Grid2Dcpp.Lsr2d(Tx, Rx, xg, zg)
l_rai = Ldroit.sum(axis=1)
lent_app = dobs/l_rai
m0 = np.mean(lent_app) + np.zeros((nx*nz,))
```

Régularisation – Exemple

- Pour les rais courbes, le calcul se fait en appelant

```
t0 = np.zeros((Tx.shape[0], ))
d, L = g.raytrace(m, [], [], Tx, Rx, t0, 2)
```

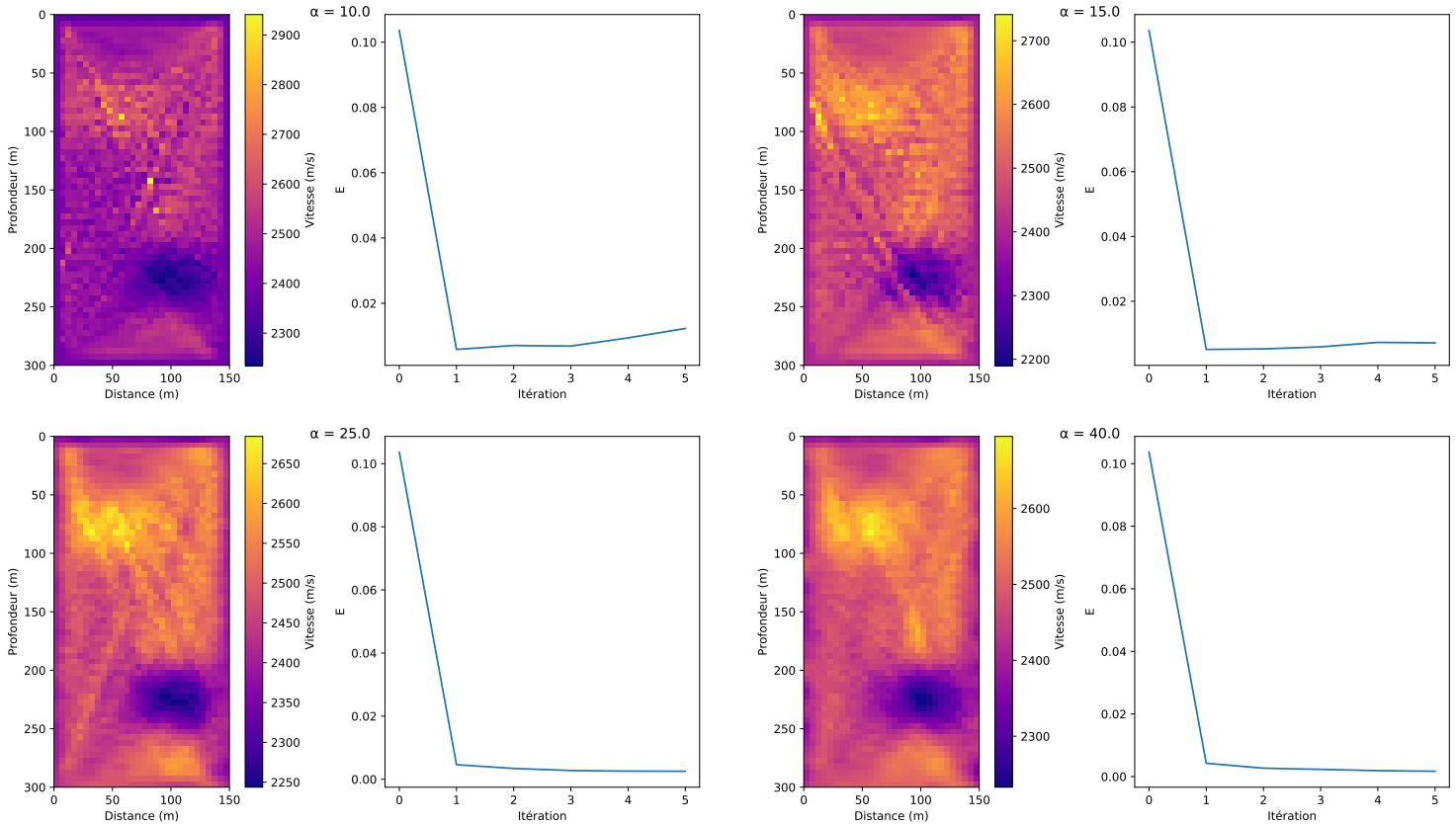
- Examinez les résultats pour 5 itérations pour les valeurs suivantes de α

```
alpha_val = [10.0, 15.0, 25.0, 40.0, 80.0, \
160.0, 300.0, 500.0, 1000.0, 10000.]
```

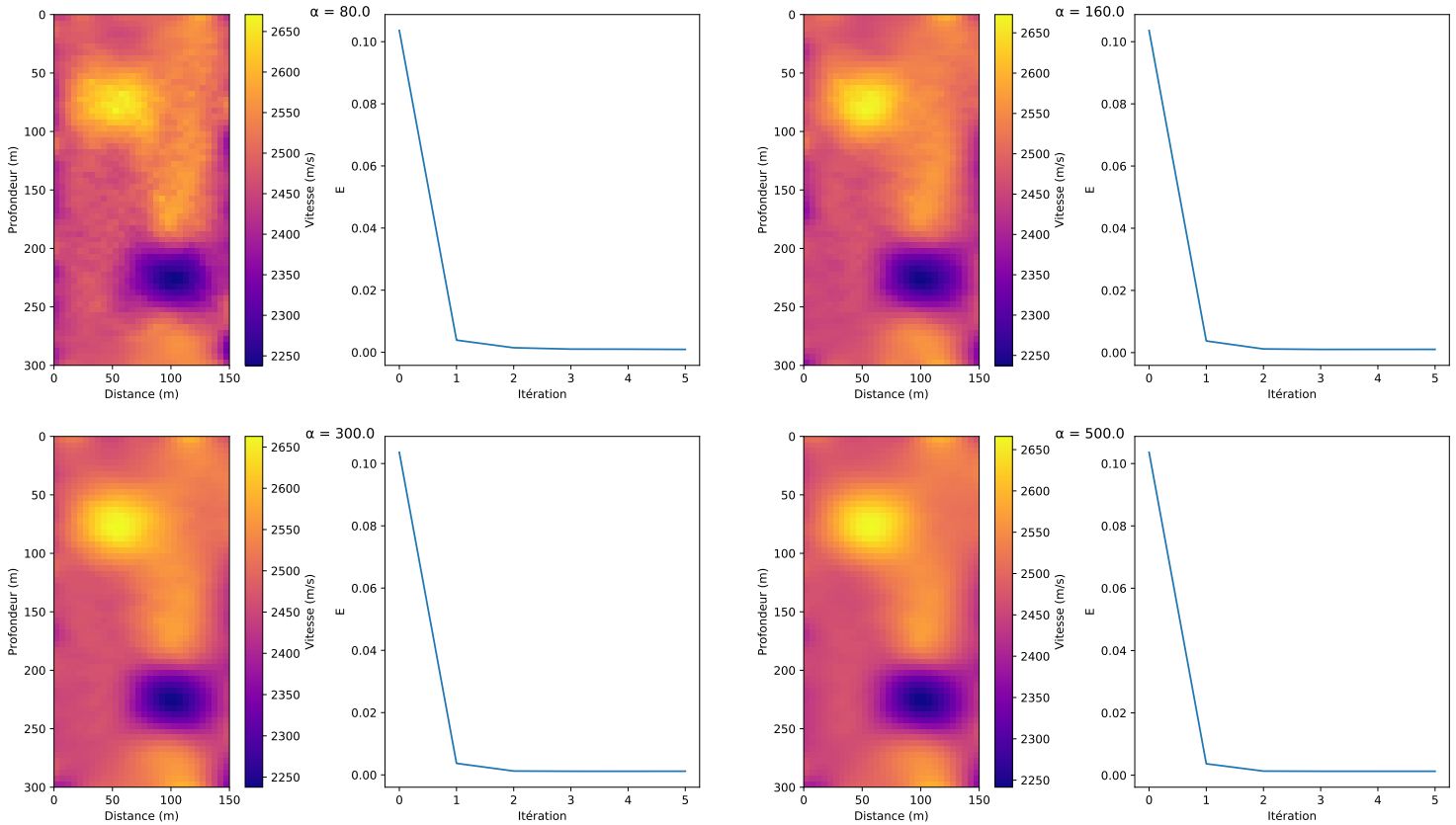
- La fonction `derivative` peut être utilisée pour calculer la matrice de lissage :

```
Dx, Dz = derivative(nx, nz, dx, dz, 2)
D = Dx + Dz
```

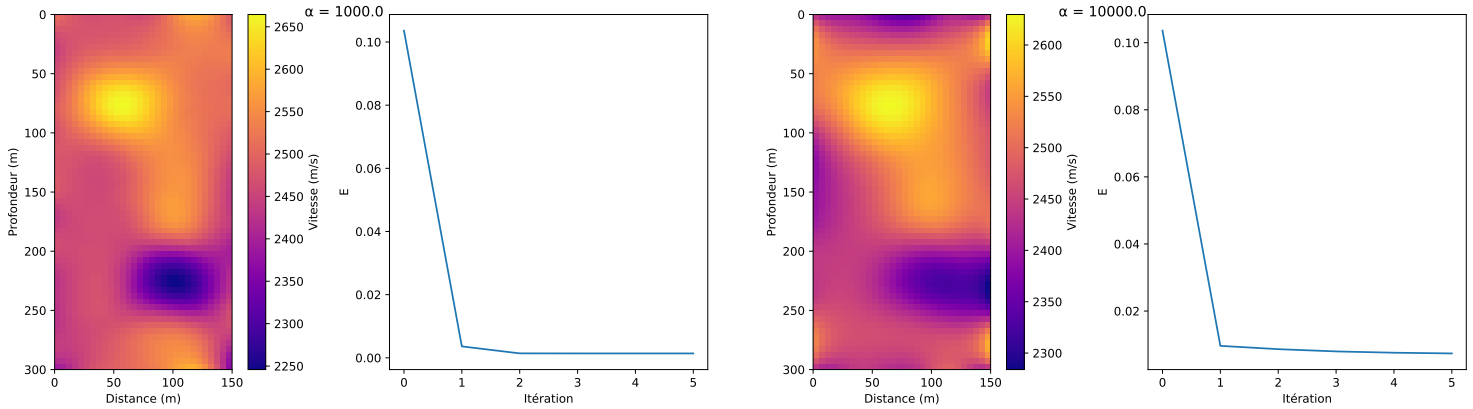
Régularisation – Exemple



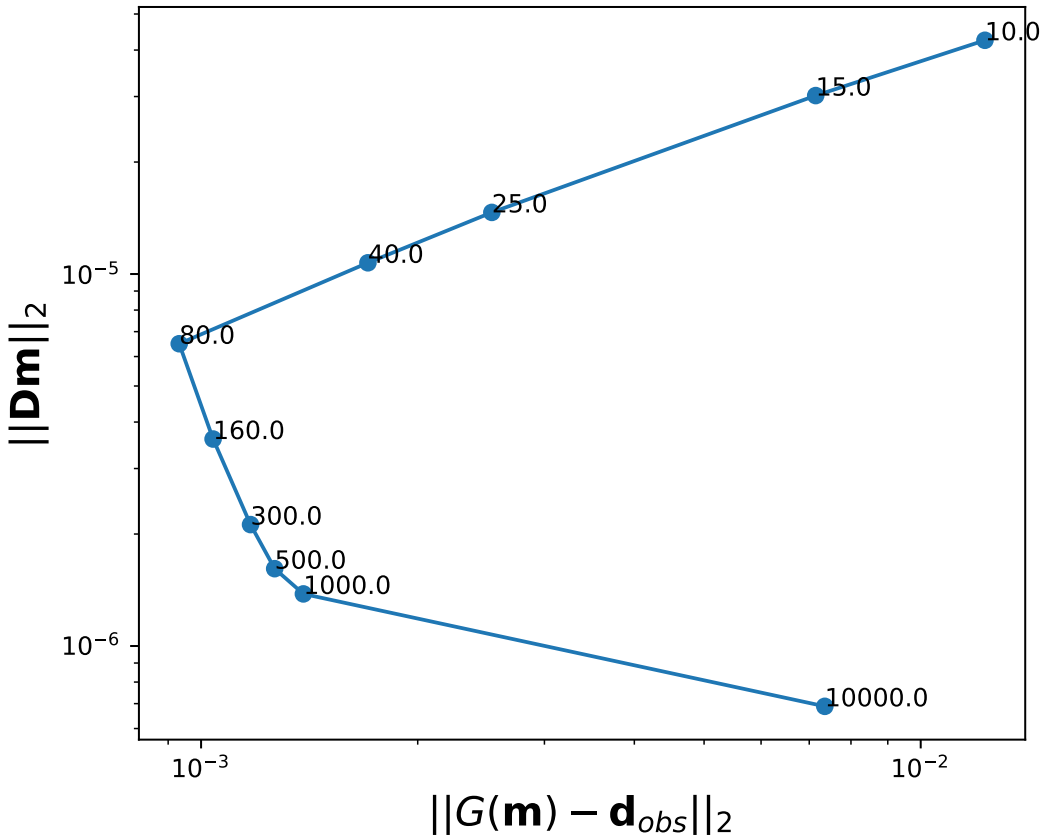
Régularisation – Exemple



Régularisation – Exemple



Régularisation – Exemple



Résolution

- On a vu en inversion linéaire que la matrice de résolution des paramètres est donnée par

$$\mathbf{m} = \mathbf{G}^{\dagger}\mathbf{G}\mathbf{m}_{\text{vrai}} = \mathbf{R}_{\text{m}}\mathbf{m}_{\text{vrai}} \tag{49}$$

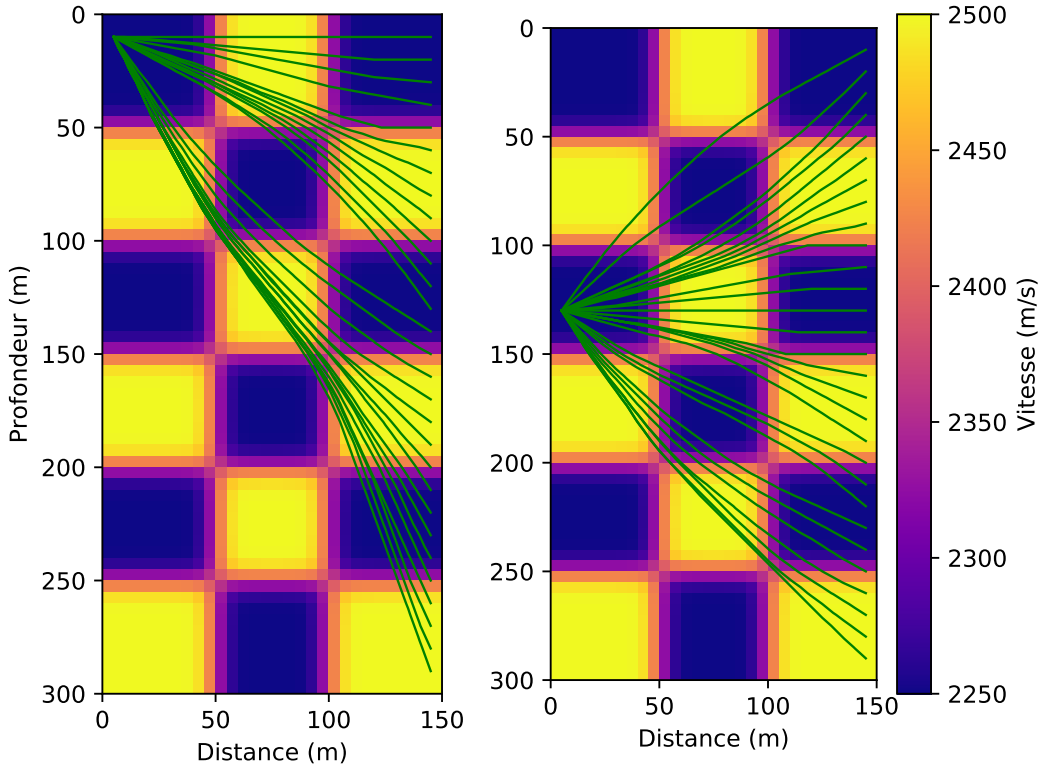
- Pour le cas non linéaire, nous aurions

$$\mathbf{m} = G^{-1}\left(G\left(\mathbf{m}_{\text{vrai}}\right)\right) \tag{50}$$

  où G^{-1} est l'opérateur inverse.
- Cependant, l'équation (50) ne peut pas être représenté par des matrices
  - $\mathbf{R}_{\text{m}}$ ne peut pas être calculé.
- Par ailleurs, la résolution dépend du choix du modèle initial.
- Pour ces raisons, on procède plutôt avec des tests de résolution.

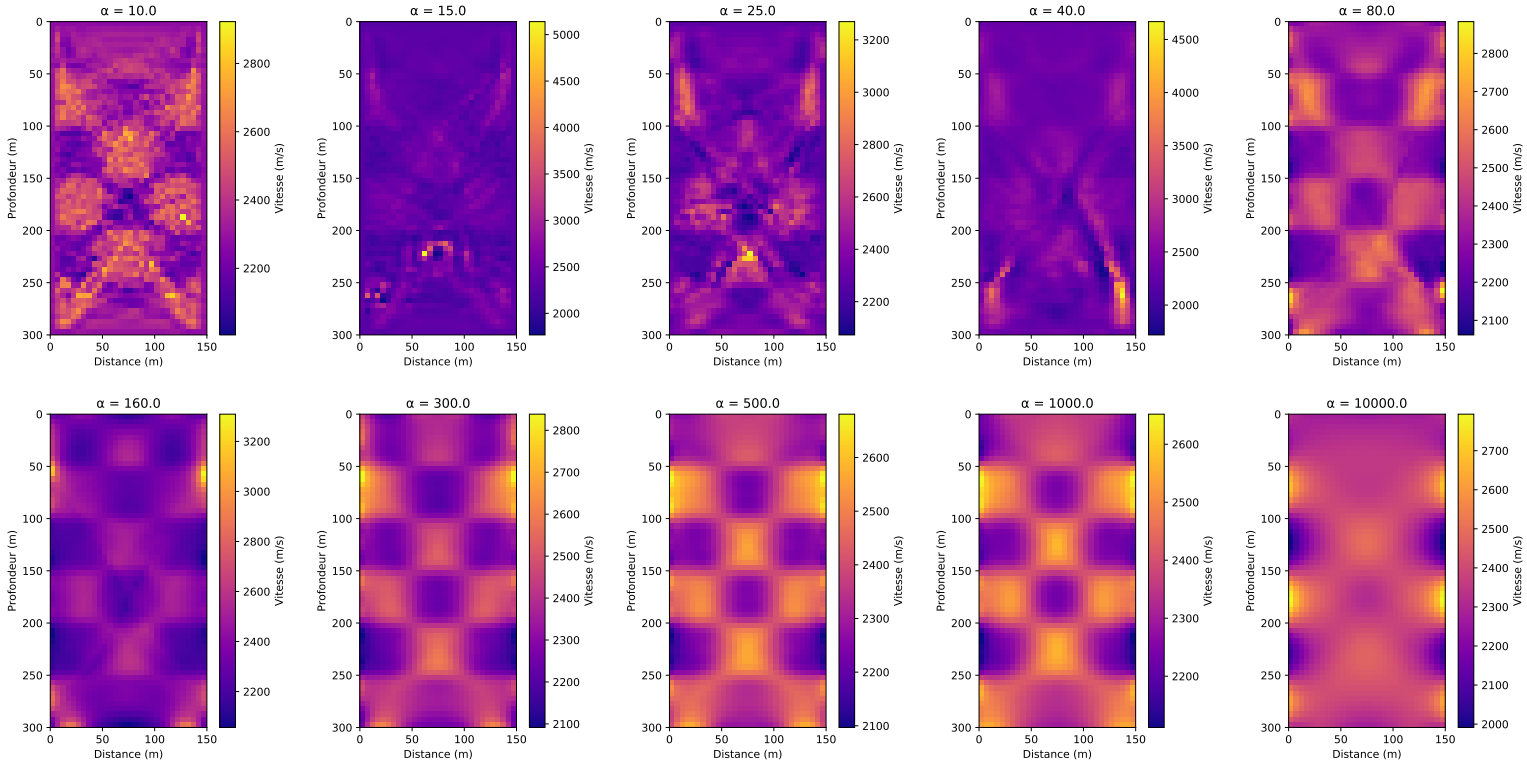
Résolution

- Le modèle en damier est souvent employé pour les tests de résolution.
- La configuration de mesure modélisée est celle qui est prévue sur le terrain.



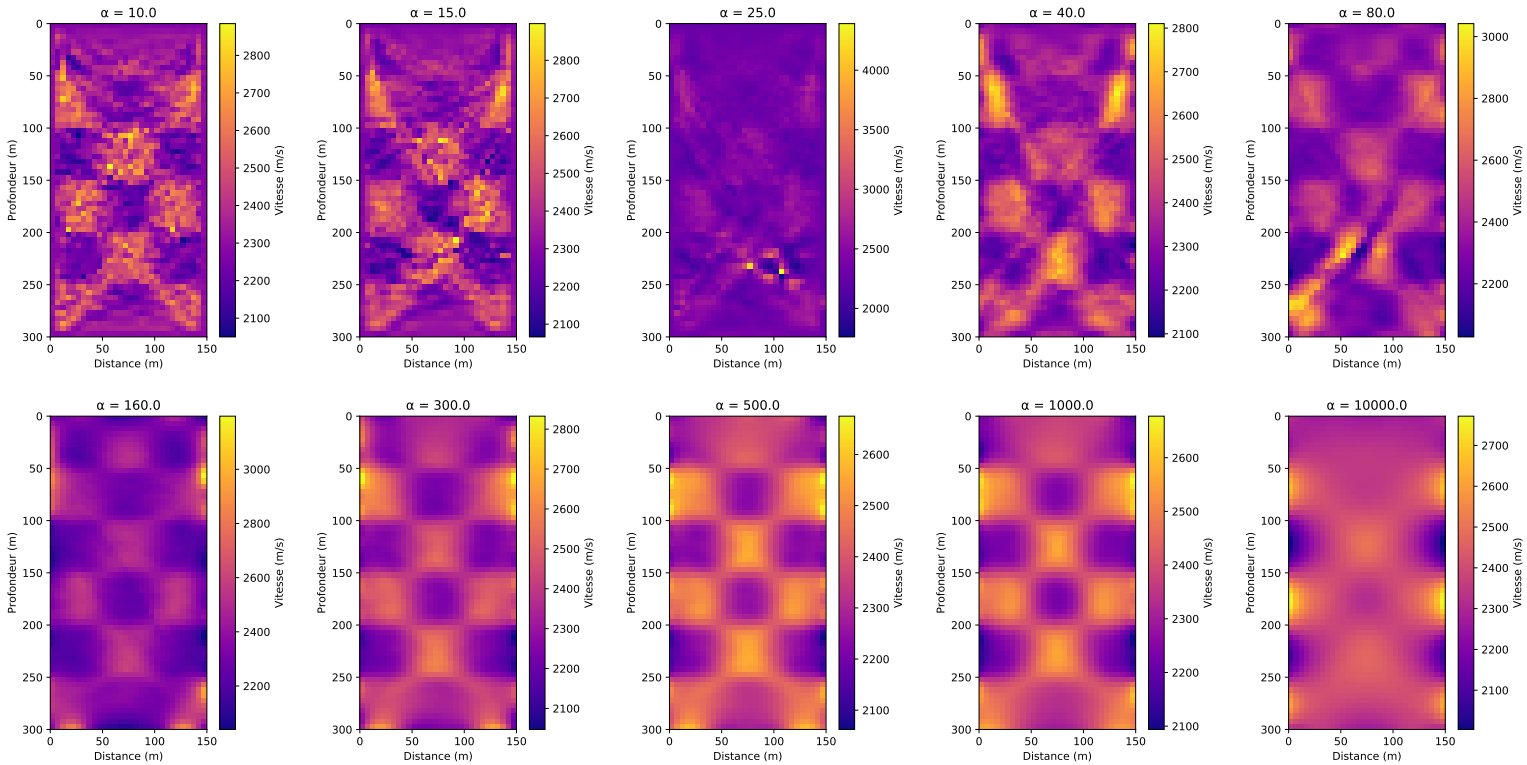
Résolution

- Résultats pour des données non bruitées.



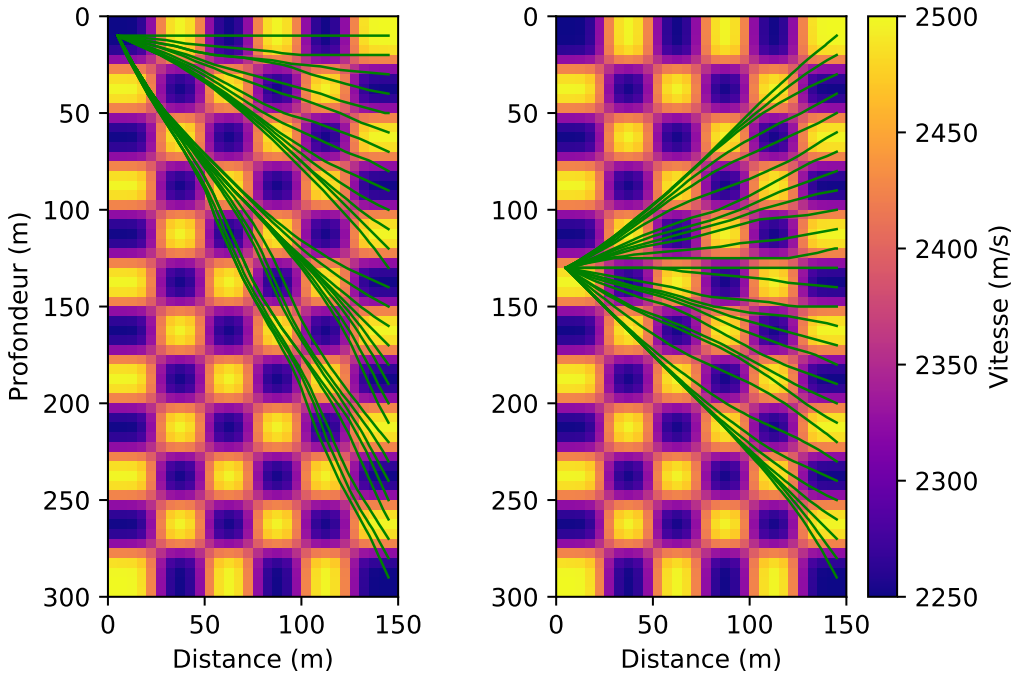
Résolution

- Résultats pour des données avec un bruit gaussien



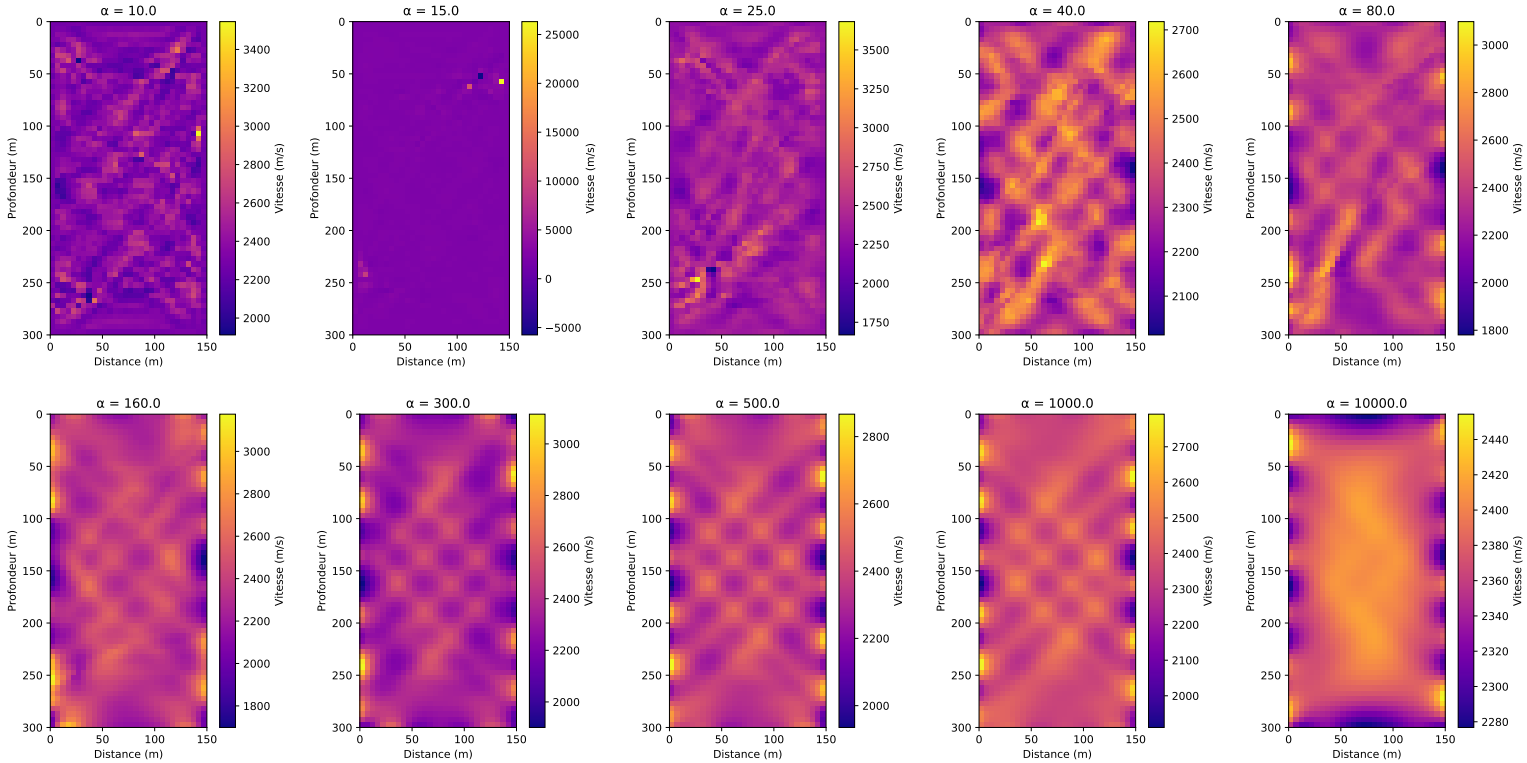
Résolution

- Des objets plus petits sont-ils résolus ?



Résolution

- Résultats pour des données non bruitées.



Résolution

- Résultats pour des données avec un bruit gaussien