Modèles spatio-dépendants en épidémiologie

Stage de première année

Titon Eve - Roynard Xavier - Fischman Sébastien

Centre de mathématiques et de leurs applications
École normale supérieure de Cachan

3 juillet 2012

Introduction

$$\boxed{S} \xrightarrow{\alpha} \boxed{I} \xrightarrow{\beta} \boxed{R}$$

$$\begin{cases} \dfrac{\partial S}{\partial t} = \nabla^2 S - \alpha S I \\ \dfrac{\partial I}{\partial t} = \nabla^2 I + \alpha S I - \beta I \\ S + I + R = 1 \\ I(0,x) \geq 0,\ S(0,x) \geq 0 \end{cases}$$

$\alpha \in \mathbb{R}^+$ coeff. de contamination

$\beta \in \mathbb{R}^+$ coeff. de récupération (immunisation)

Introduction

$$\boxed{S} \xrightarrow{\alpha} \boxed{I} \xrightarrow{\beta} \boxed{R}$$

$$\begin{cases} \dfrac{\partial S}{\partial t} = \nabla^2 S - \alpha S I \\ \dfrac{\partial I}{\partial t} = \nabla^2 I + \alpha S I - \beta I \\ S + I + R = 1 \\ I(0,x) \geq 0,\ S(0,x) \geq 0 \end{cases}$$

$\alpha \in \mathbb{R}^+$ coeff. de contamination

$\beta \in \mathbb{R}^+$ coeff. de récupération (immunisation)

Introduction

$$\boxed{S} \xrightarrow{\alpha} \boxed{I} \xrightarrow{\color{red}\beta} \boxed{R}$$

$$\begin{cases} \dfrac{\partial S}{\partial t} = \nabla^2 S - \alpha S I \\ \dfrac{\partial I}{\partial t} = \nabla^2 I + \alpha S I {\color{red}- \beta I} \\ S + I + R = 1 \\ I(0,x) \geq 0,\ S(0,x) \geq 0 \end{cases}$$

$\alpha \in \mathbb{R}^+$ coeff. de contamination

$\beta \in \mathbb{R}^+$ coeff. de récupération (immunisation)

Plan

Rappels : Théorèmes de point fixe

Lemme de Non-Rétraction

Il n'existe pas de rétraction de B^n dans S^{n-1}.
ie $\nexists f : B^n \longrightarrow S^{n-1}$ continue telle que $f_{|S^{n-1}} = Id$

Théorème de Brouwer

Soit $f : B^n \longrightarrow B^n$ continue, f admet un point fixe.

Théorème de Schauder

Soient $(E, ||.||)$ un espace de Banach et C convexe fermé non vide de E.
Toute application $f : C \longrightarrow C$ continue telle que $\overline{f(C)}$ est compact a un point fixe.

Méthode générale pour montrer l'existence de solutions

- On s'intéresse ici à l'existence de solutions stationnaires du système précedent, c'est-à-dire sans terme temporel :

$$\begin{cases} -s'' = -\alpha si \quad et \quad s(a) = \alpha_1,\ s(b) = \beta_1 \\ -i'' = \alpha si - \beta i \quad et \quad i(a) = \alpha_2,\ i(b) = \beta_2 \end{cases}$$

- On part d'un système d'EDP:

$$\begin{cases} -u'' = f(u,v) \quad et \quad u(a) = \alpha_1,\ u(b) = \beta_1 \\ -v'' = g(u,v) \quad et \quad v(a) = \alpha_2,\ v(b) = \beta_2 \end{cases}$$

- Que l'on transforme en système intégral équivalent:

$$\begin{cases} \forall x \in [a,b],\ u(x) = \alpha_1 + (x-a).\left(\dfrac{\beta_1 - \alpha_1 + \int_a^b \int_a^t f(u(s),v(s))dsdt}{b-a}\right) - \int_a^x \int_a^t f(u(s),v(s))dsdt = F(u,v)(x) \\ \forall x \in [a,b],\ v(x) = \alpha_2 + (x-a).\left(\dfrac{\beta_2 - \alpha_2 + \int_a^b \int_a^t g(u(s),v(s))dsdt}{b-a}\right) - \int_a^x \int_a^t g(u(s),v(s))dsdt = G(u,v)(x) \end{cases}$$

- Ce qui nous donne un problème de point fixe de (F, G).

Vérification des hypothèses du Théorème de Schauder et choix d'un bon convexe

1. On prend le convexe K^2 avec $K = \{u \geqslant 0, \|u\|_{L^2(]a,b[)} \leqslant C\}$, et on cherche C tel que $(F,G)(K^2) \subset K^2$
2. Inégalité de Poincaré :

Proposition

Soit $f \in C^1([a,b])$ vérifiant les conditions de Dirichlet homogènes(c'est-à-dire $f(a) = f(b) = 0$),
Alors

$$\|f\|_{L^2(]a,b[)} \leqslant (b-a).\|f'\|_{L^2(]a,b[)}$$

3. Ceci nous permet de trouver $C = (b-a)^{\frac{5}{2}} \max(\|f\|_{L^\infty}, \|g\|_{L^\infty}) + max(|\alpha_1|, |\beta_1|, |\alpha_2|, |\beta_2|)\sqrt{b-a}$

Approximation de notre système pour rentrer dans les hypothèses

1. On approxime notre système :

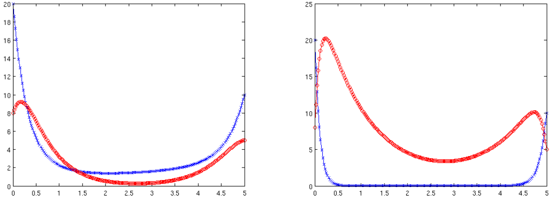
$$\begin{cases} -s_\epsilon'' = -\alpha \dfrac{s_\epsilon}{1+\epsilon s_\epsilon^2} \dfrac{i_\epsilon}{1+\epsilon i_\epsilon^2} \quad \textit{et} \quad s_\epsilon(a) = \alpha_1,\ s_\epsilon(b) = \beta_1 \\ -i_\epsilon'' = \alpha \dfrac{s_\epsilon}{1+\epsilon s_\epsilon^2} \dfrac{i_\epsilon}{1+\epsilon i_\epsilon^2} - \beta \dfrac{i_\epsilon}{1+\epsilon i_\epsilon^2} \quad \textit{et} \quad i_\epsilon(a) = \alpha_2,\ i_\epsilon(b) = \beta_2 \end{cases}$$

2. Par Théorème d'Ascoli, on extrait une valeur d'adhérence (s, i) de (s_ϵ, i_ϵ) quand $\epsilon \longrightarrow 0$
3. On vérifie que (s, i) est solution de notre système.

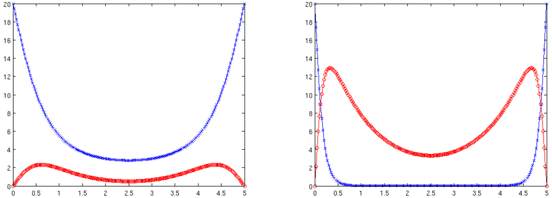
Simulation numérique

Grâce à un schéma aux différences finies, on résoud le système :

$$\begin{cases} -s'' = -\alpha si \quad et \quad s(a) = \alpha_1,\ s(b) = \beta_1 \\ -i'' = \alpha si - \beta i \quad et \quad i(a) = \alpha_2,\ i(b) = \beta_2 \end{cases}$$



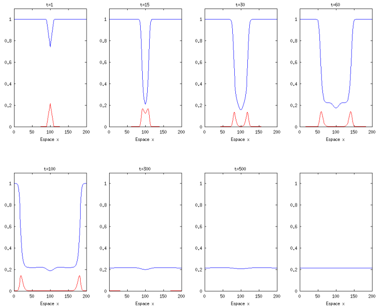
À gauche $\beta < \alpha$, à droite $\beta \geq \alpha$.



Dans le cas où on met en quarantaine les infectés aux limites.À gauche $\beta < \alpha$, à droite $\beta \geq \alpha$.

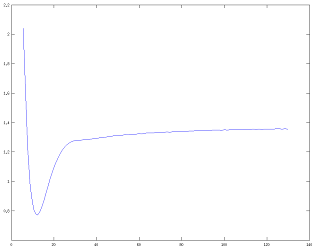
Fronts progressifs

- Condition nécessaire et suffisante pour l'exitence d'un front progressif: $\alpha > \beta$



- Trois phases majeures:
  1. Relaxation de la condition initiale (diminution et mise en place d'un front)
  2. Propagation d'un front
  3. Homogénéisation de la population

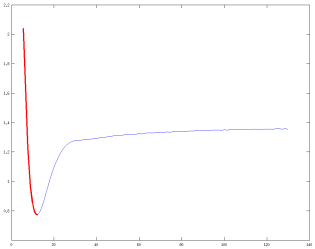
Vitesse des fronts progressifs



Vitesse en fonction du temps, pour le front de droite, obtenue pour $\alpha = 1.5$ et $\beta = 1$

- diminution de la condition initiale
- établissement du front
- la propagation du front

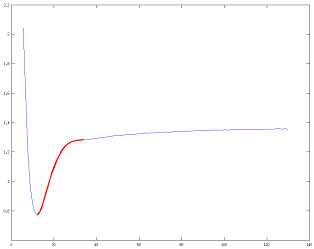
Vitesse des fronts progressifs



Vitesse en fonction du temps, pour le front de droite, obtenue pour $\alpha = 1.5$ et $\beta = 1$

- diminution de la condition initiale
- établissement du front
- la propagation du front

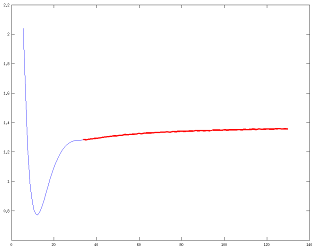
Vitesse des fronts progressifs



Vitesse en fonction du temps, pour le front de droite, obtenue pour $\alpha = 1.5$ et $\beta = 1$

- diminution de la condition initiale
- établissement du front
- la propagation du front

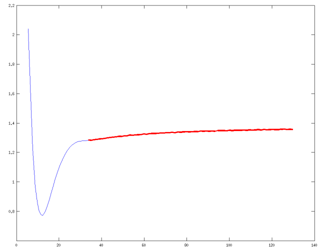
Vitesse des fronts progressifs



Vitesse en fonction du temps, pour le front de droite, obtenue pour $\alpha = 1.5$ et $\beta = 1$

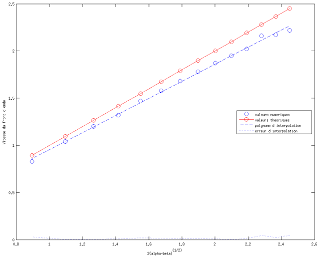
- diminution de la condition initiale
- établissement du front
- la propagation du front

Vitesse des fronts progressifs



Vitesse en fonction du temps, pour le front de droite, obtenue pour $\alpha = 1.5$ et $\beta = 1$

- diminution de la condition initiale
- établissement du front
- la propagation du front

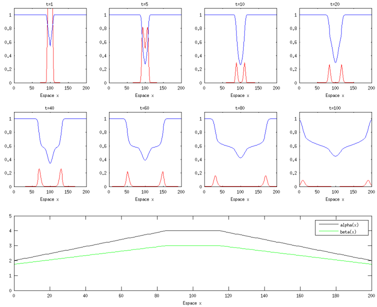


Vitesse de propagation du front tracée en fonction de $2(\alpha - \beta)^{\frac{1}{2}}$

- Vitesse théorique: $V = 2(\alpha - \beta)^{\frac{1}{2}}$ [Murray, 2009]
- Valeurs obtenues interpolées au sens des moindres carrés par: $y = 0.9051x + 0.0497$

Coefficients variables: influence sur les fronts progressifs

- Coefficients $\alpha(x)$, $\beta(x)$, non constants, tels que $\forall x,\ \alpha(x) > \beta(x) \geq 0$.
- Avec des coefficients affines



- Existence de fronts progressifs se déformant.

Conclusion

- Démonstration d'existence de solutions
- Simulations numériques : des résultats contre-intuitifs
- Perspectives liées aux observations

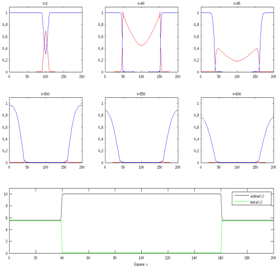
Bibliographie

A. Galvani, J. Novembre, and M. Slatkin.
The geographic spread of the ccr5 δ 32 hiv-resistance allele.
2005.

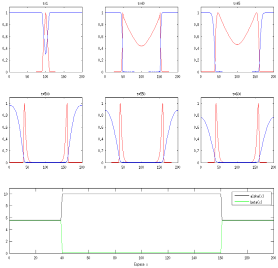
M. Ghassani, J. Gaudart, J. Minsta, M. Rachdi, J. Waku, and J. Demongeot.
Demography and diffusion in epidemics: Malaria and death spread.
2010.

J.D. Murray.
Mathematical Biology, volume 2.
Springer, 2009.

Comment stopper une épidémie?



courbes bleues: S; courbes rouges: $10 * I$



courbes bleues: S; courbes rouges: $I / max(I)$

Diminution importante et brusque de $\alpha(x) - \beta(x)$ $\Rightarrow$ arrêt de l'épidémie.

Active Experimental Design for Tsunami Modeling

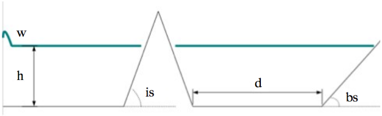
Chloé Pasin - Maxime Stauffert - Alexandre Robicquet

Encadrants : Nicolas Vayatis - David Buffoni - Themistoklis Stefanakis

3 Juillet 2012

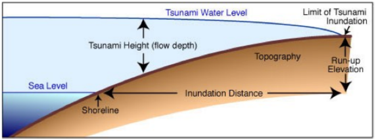
Modélisation et simulation d'un tsunami

- 5 paramètres physiques



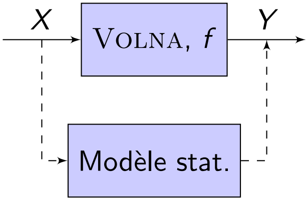
is	0.05-0.2
bs	0.05-0.2
d	0 - 5000m
h	100 - 1000m
w	0.01 - 0.1 rad/s

- "single wave" : $\eta(x_0, t) = A_0 sech^2(wt - 2.6)$ avec $A_0 = 1.5m$
- Etude de l'amplification du run-up en un point sur la plage derrière l'île par rapport à un point latéral indépendant de l'île [Briggs et al., 1995]



- Modélisation des tsunamis grâce au code VOLNA (déterministe) [Dutykh, Poncet, Dias, 2011]

Problématique



- Objectif : $X^* = \underset{X \in \mathcal{X}}{\operatorname{argmax}} (f(X))$
- Difficultés :
  - f inconnue, sauf sur un échantillon donné $(X_1, .., X_n)$
  - calcul de $f(x)$ très coûteux (dans notre cas, 60-90 min)
- Démarche :
  - Plan d'expérience séquentiel
  - Techniques d'apprentissage actif

Protocole expérimental

- 30 points étiquetés sur les 200 obtenus
- Question : exploration des autres points
- Evaluation de la performance de la méthode :

Type de méthode	Type d'erreur
Régression $Y \in \mathbb{R}$	Erreur quadratique moyenne : $\sum_{i=1}^{n}(Y_i - f(X_i))^2$
Classification $Y \in \{-1, 1\}$	Taux d'erreur : $\frac{1}{n}\sum_{i=1}^{n}\mathbb{1}_{f(X_i)\neq Y_i}$
Ranking $Y \in \mathfrak{S}$	Discounted Cumulated Gain : $DCG(Y, s) = \sum_{i=1}^{n}\frac{2^{Y_i}-1}{ln(1+i)}$ s : ordonne les $f(X_i)$ dans l'ordre décroissant

Modèle : $Y = X\theta + \epsilon$ avec $\epsilon \sim N(0, \sigma^2 I_n)$, indépendant de X.

- **Méthode linéaire** [Hastie, Tibshirani, Friedman, 2008]
  Estimateur Ridge :

$$\widehat{\theta} = \underset{\theta \in \mathbb{R}^d}{\operatorname{argmin}} ||Y - \tilde{X}\theta||^2 + \lambda ||\theta||^2$$

- **Méthode non linéaire** : Kernels
  Kernel polynomial de degré d : $\tilde{X}_{i,j} = ({}^tXX)^d_{i,j}$
  Kernel gaussien : $\tilde{X}_{i,j} = e^{-\frac{||X_i - Xj||^2}{2\sigma^2}}$

Apprentissage actif

- Principe de l'apprentissage actif : sélection des points les plus pertinents.
- Intérêt : temps computationnel diminué + convergence vers le prédicteur cible
- Majorité des travaux en actif réduit à de la classification [Yu et al., 2006]

Différentes approches :
- Query By Committee (ex : CAL)
- Uncertainty Sampling
- Expected Model Change

Apprentissage actif pour la classification [Hsu, 2010]

- Algorithme CAL [Cohn, Atlas, Ladner, 1994] :
  - un exemple inconnu nous est fourni
  - on apprend deux modèles ($y = +1$ et $y = -1$)
  - si on prédit le même label $\rightarrow$ on infère
  - sinon on demande (région d'incertitude)

Algorithme 1: CAL

Input : $Z_0 = \varnothing$, $\mathcal{V}_0 = \mathcal{H}$

for t = *1,2,...,T* **do**

  Tirer X_t;

  if $\exists h, h' \in \mathcal{V}_{t-1}$ *tels que* $h(X_t) \neq h'(X_t)$;

  then

    Demander Y_t et poser $Z_t := Z_{t-1} \cup (X_t, Y_t)$;

  else

    Poser $\tilde{Y}_t := h(X_t), \forall h \in \mathcal{V}_{t-1}$ et $Z_t := Z_{t-1} \cup (X_t, \tilde{Y}_t)$;

  Poser $\mathcal{V}_t := \{h \in \mathcal{H} / h(X_i) = Y_i, \forall (X_i, Y_i) \in Z_t\}$;

Output : Tous les $h \in \mathcal{V}_T$

- Algorithme Phased CAL [Hsu, 2010] :

  - On change de phase lorsque

$$\mathbb{P}(X \in \mathcal{R}(\mathcal{V}_t)) \leq \frac{1}{2}\mathbb{P}(X \in \mathcal{R}(\mathcal{V}_{t_p}))$$

  - Coefficient de désaccord [Hanneke, 2007] :

$$\theta(\mathcal{H},\mathcal{D}) := \sup\left\{\frac{\mathbb{P}(X \in \mathcal{R}(\mathcal{B}(h^*, r))}{r}\ ,\ r > 0\right\}$$

Théorème

$\forall \epsilon, \delta \in]0;1[$, avec probabilité d'au moins $1-\delta$, Phased CAL retourne une hypothèse $h \in \mathcal{H}$ qui a une erreur inférieure à ϵ après avoir demandé au plus $O\left(\theta(\mathcal{H},\mathcal{D})\left(\ln\left(\frac{|\mathcal{H}|}{\delta}\right) + \ln\left(\ln\left(\frac{1}{\epsilon}\right)\right)\right)\ln\left(\frac{1}{\epsilon}\right)\right)$.

Apprentissage actif pour la classification [Hsu, 2010]

- Algorithme DHM [Dasgupta, Hsu, Monteleoni,2007] :
  - amélioration de CAL (cas région d'incertitude)
  - $\Delta =$ différence des erreurs entre les deux modèles
  - si $|\Delta|$ suffisamment grand $\rightarrow$ on infère
  - sinon on demande

- Algorithme IWAL [Beygelzimer, Dasgupta, Langford, 2009] :
  - un exemple inconnu nous est fourni
  - $p_t =$ probabilité de demander son label
  - si demandé, ajout à l'ensemble des points connus avec un poids $\propto 1/p_t$

$\rightarrow$ permet d'éviter le biais d'échantillonnage [Dasgupta, 2011]

Apprentissage actif pour la régression

- Lacune des algorithmes précédents : appliqués à la classification binaire
- Intérêt : adaptation immédiate au problème (régression)

Contributions du stage

- Sélection des points à chaque itération :
  - ARGMAX

$$X = \underset{X_i \in X}{\operatorname{argmax}}(\hat{Y}_i)$$

  - MAXLIPSCHITZ

$$\mathcal{V}_n = \{f \text{ k-lipschitz} \mid f(x_i) = y_i, \forall (x_i, y_i) \in \mathcal{B}\}$$

$$X = \underset{x \in X}{\operatorname{argmax}} \max_{f \in \mathcal{V}_n} f(x)$$

- Critère d'arrêt : Inspiré du DCG, évaluation du ranking de l'estimateur $\hat{\theta}$ sur une base annexe.

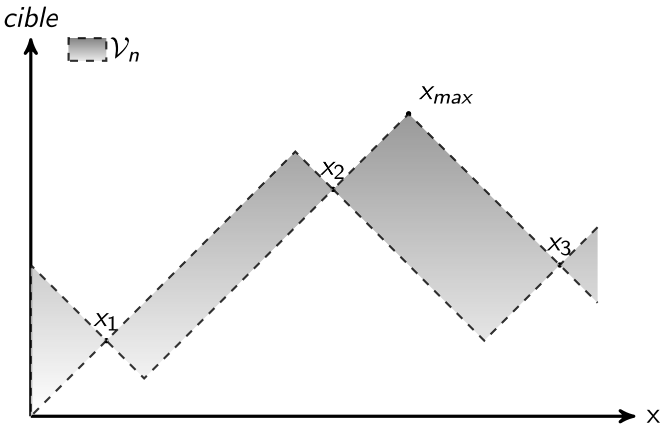


FIGURE: MaxLipschitz sur $\mathcal{S} = \{x_1, x_2, x_3\}$

$$\mathcal{V}_n = \{f \text{ k-lipschitz} \mid f(x_i) = y_i, \forall (x_i, y_i) \in \mathcal{B}\}$$

$$X = \underset{x \in X}{\operatorname{argmax}} \max_{f \in \mathcal{V}_n} f(x)$$

Algorithme 2: StoppingCriterion

Input : $(\hat{\theta}_{n+1}, \hat{\theta}_n)$
Input : $\Gamma \subset X$ base de validation utilisée pour le critère d'arrêt
$r_n = rank(\langle\hat{\theta}_n, \Gamma\rangle)$;
$r_{n+1} = rank(\langle\hat{\theta}_{n+1}, \Gamma\rangle)$;
$E_{n+1} = \sum_{j=1}^{k} \frac{|r_{n+1,i} - r_{n,i}|^2}{r_i^2}$;
Output : E_{n+1}

But : trouver un seuil ϵ :

$$si \quad E_{n+1} < \epsilon \quad \longrightarrow \quad stop\ algorithme$$

Expériences : classification sur données réelles

TABLE: Tableau des résultats sur la base *Tsunamis* convertie en un problème de classification avec apprentissage en régression Ridge seuillée (15% de positifs et 85% de négatifs). Erreur pondérée et moyennée sur 250 expériences.

Algorithme	Erreur	NDCG	NDCG@10	**Nb de points demandés**
Témoin	0.374	0.756	0.505	**100%**
CAL	0.398	0.663	0.393	**37.5%**
DHM	0.394	0.721	0.465	**37.5%**
IWAL	0.382	0.695	0.407	**23.62%**

Expériences : classification sur données simulées

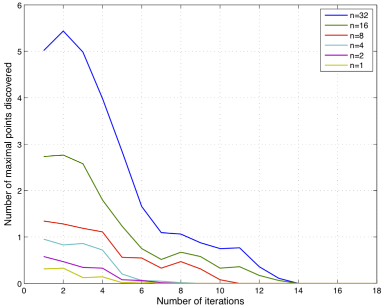
TABLE: Tableau des résultats sur la base *Gaussienne* convertie en un problème de classification avec apprentissage en régression Ridge seuillée (10% de positifs et 90% de positifs). Erreur pondérée et moyennée sur 250 expériences.

Algorithme	Erreur	NDCG	NDCG@10	**Nb de points demandés**
Témoin	0.307	0.807	0.685	**100%**
CAL	0.400	0.645	0.432	**11.6%**
DHM	0.343	0.706	0.422	**9.96%**
IWAL	0.332	0.724	0.465	**8.3%**

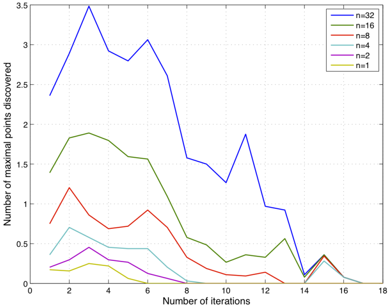
- Hiérarchie entre les algorithmes d'apprentissage actif
- Gain de temps pour performances relativement égales

Expériences : régression sur données réelles

Test des algorithmes de sélection sur la base *Tsunamis*.
Evaluation du nombre de maxima locaux découverts en fonction du nombre d'itérations.



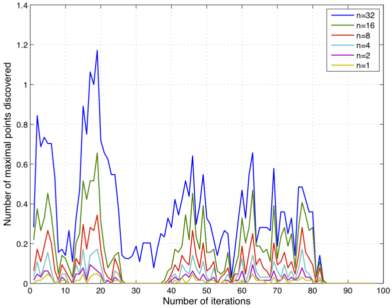
(a) ARGMAX



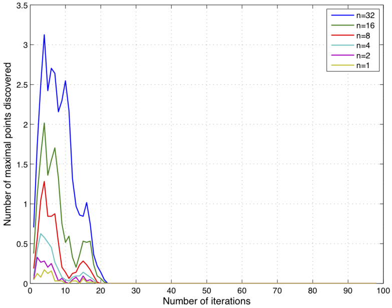
(b) MAXLIPSCHITZ

Expériences : régression sur données simulées

Test des algorithmes de sélection sur la base *Gaussienne*.
Evaluation du nombre de maxima locaux découverts en fonction du nombre d'itérations.



(c) ARGMAX



(d) MAXLIPSCHITZ

Conclusions et Perspectives

Conclusions

- Étude approfondie de l'apprentissage actif en classification.
- Réalisation d'un algorithme actif en régression.
  - Sélection des points (MaxLipschitz).
  - Critère d'arrêt de l'algorithme (basé sur le DCG).

Perspectives

- Application réelle avec Volna (nombre indéfini de simulations).
- Seuillage dynamique (passage de la régression à la classification).
- Résultats théoriques sur l'algorithme actif en régression (vitesse de convergence, complexité...).

Bibliographie

[Briggs et al., 1995] Michael J. Briggs, Costas E. Synolakis, Gordon S. Harkins, and Debra R. Green. Laboratory Experiments of Tsunami Runup on a Circular Island. *Pure and Applied Geophysics* , 1995.

[Dutykh, Poncet, Dias, 2011] Denis Dutykh, Raphaël Poncet, and Frederic Dias. The VOLNA code for the numerical modeling of tsunami waves : Generation, propagation and inundation. *European Journal of Mechanics - B/Fluids*, 30(6) :598-615, 2011.

[Hastie, Tibshirani, Friedman, 2008] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2008.

[Yu et al., 2006] Kai Yu, Jinbo Bi, and Volker Tresp. Active learning via transductive experimental design. *Proceedings of the 23rd international conference on Machine learning*, 2006.

[Hsu, 2010] Daniel Joseph Hsu. *Algorithms for active learning*. Phd thesis, La Jolla, CA, USA, 2010.

[Dasgupta, 2011] Sanjoy Dasgupta. Two faces of active learning. *Theor. Comput. Sci.*, 412 :1767-1781, 2011.

Etude de la convergence de schémas aux volumes finis

P., J.-C. Bertrand et J. Guérand

Sous la responsabilité de D. Bouche et F., P. Pascal

3 Juillet 2012

Plan

Le théorème de Lax-Richtmyer

u solution exacte de :

$$\mathrm{L(u) = F}$$

En discrétisant :

$$L_h(u_h) = F$$

Définition

Erreur de consistance : $\epsilon_h = L_h(u) - F$

Théorème

Consistance + Stabilité ⇒ Convergence

Stabilité

$$|||L_h^{-1}||| \leq c$$

Ordre de consistance p :

$$\epsilon_h = \mathcal{O}(h^p)$$

$$||e_h|| = ||u - u_h|| = ||L_h^{-1}(L_h(u - u_h))|| \leq c||\epsilon_h|| = \mathcal{O}(h^p)$$

Supra-convergence

Définition

Un schéma est supra-convergent lorsque e_h a un ordre de convergence supérieur à ϵ_h.

C.S. de supra-convergence : Existence de γ tel que :

- $\gamma = \mathcal{O}(h^p)$
- $\epsilon_h = L_h(\gamma) + \underline{\epsilon}_h = \mathcal{O}(h^{p-1})$
- $\underline{\epsilon}_h = \mathcal{O}(h^p)$

$$\Rightarrow L_h(e_h - \gamma) = \underline{\epsilon}_h$$

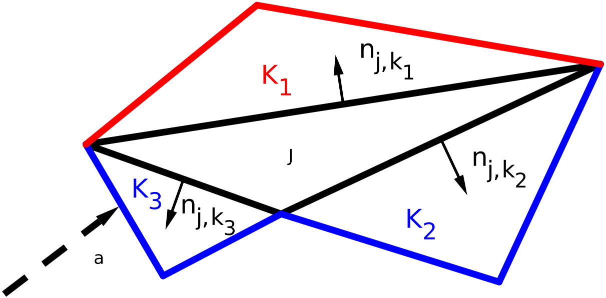
$$\Rightarrow ||e_h - \gamma|| = \mathcal{O}(h^p)$$

$$\Rightarrow ||e_h|| = \mathcal{O}(h^p)$$

Equation de transport et notations

$$\frac{\partial u}{\partial t} + (a.\nabla)u = 0, \qquad (x,t) \in \Omega \times]0,+\infty[$$

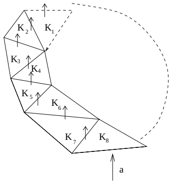
$$\frac{u_j^{n+1} - u_j^n}{\Delta t_n} + \frac{1}{|K_j|}\Big(\sum_{k\in\mathcal{N}^+(j)} a.N_{j,k}u_j^n + \sum_{k\in\mathcal{N}^-(j)} a.N_{j,k}u_k^n\Big) = 0$$

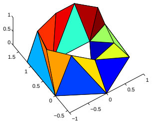


Rappels de résultats déja exposés

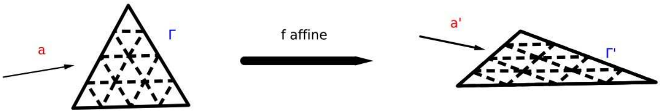
$$(I + B)\Gamma = \Delta$$

- B est nilpotente en 2D mais pas en 3D





- Calcul explicite du correcteur dans le cas d'un triangle quelconque raffiné uniformément



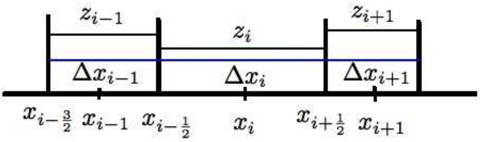
Equation d'advection linéaire avec terme source

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} + b(u)z'(x) = 0, \qquad a > 0$$

Schéma (non linéaire) étudié

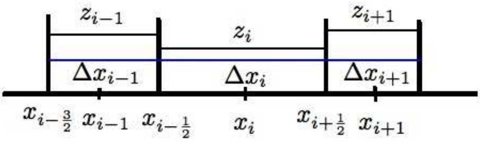
$$S(v) = \frac{v_i^{n+1} - v_i^n}{\Delta t} + a\frac{v_i^n - v_{i-1}^n}{\Delta x_i} + \underbrace{b(v_i^n)\frac{z(x_i) - z(x_{i-1})}{\Delta x_i}}_{\Sigma_i^n}$$

avec v_j^n approximation de $\frac{\int_{K_j} u(x,t^n)dx}{|K_j|}$.



Que choisir pour Σ_i^n ?

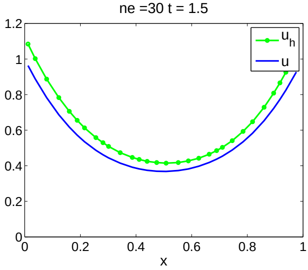
- $\Sigma_{i,1}^n = b(v_i^n)\frac{z(x_i)-z(x_{i-1})}{\Delta x_i}$
- $\Sigma_{i,2}^n = b(v_i^n)\frac{z(x_{i+1})-z(x_i)}{\Delta x_i}$
- $\Sigma_{i,3}^n = b(v_i^n)z'(x_i)$
- $\Sigma_{i,4}^n = \frac{1}{\Delta x_i}\left(\Delta x_{i-1} b(v_{i-1}) + \Delta x_i b(v_i)\right)\frac{z(x_i)-z(x_{i-1})}{\Delta x_i+\Delta x_{i-1}}$



Résultats numériques pour $\Sigma_{i,1}^n$, $\Sigma_{i,2}^n$ et $\Sigma_{i,3}^n$

$$z(x) = sin(\pi x),\ b(x) = x,\ a = 1$$

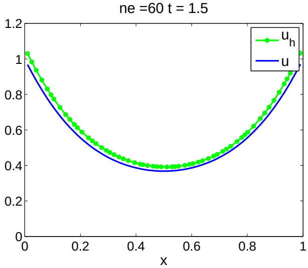
Conditions aux bords périodiques



Résultats numériques pour $\Sigma_{i,1}^n$, $\Sigma_{i,2}^n$ et $\Sigma_{i,3}^n$

$$z(x) = sin(\pi x),\ b(x) = x,\ a = 1$$

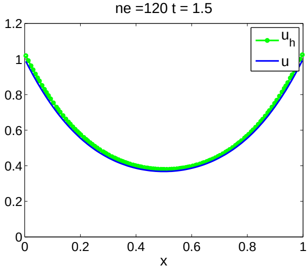
Conditions aux bords périodiques



Résultats numériques pour $\Sigma_{i,1}^n$, $\Sigma_{i,2}^n$ et $\Sigma_{i,3}^n$

$$z(x) = sin(\pi x),\ b(x) = x,\ a = 1$$

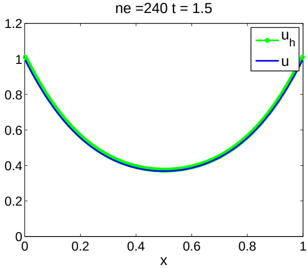
Conditions aux bords périodiques



Résultats numériques pour $\Sigma_{i,1}^n$, $\Sigma_{i,2}^n$ et $\Sigma_{i,3}^n$

$$z(x) = sin(\pi x),\ b(x) = x,\ a = 1$$

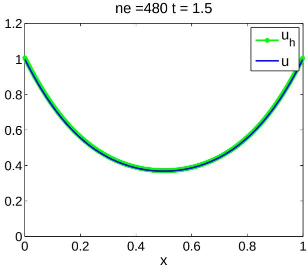
Conditions aux bords périodiques



Résultats numériques pour $\Sigma_{i,1}^n$, $\Sigma_{i,2}^n$ et $\Sigma_{i,3}^n$

$$z(x) = sin(\pi x),\ b(x) = x,\ a = 1$$

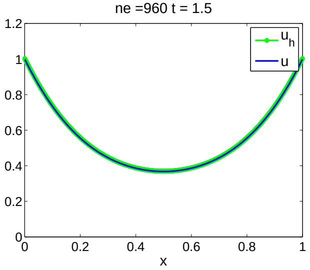
Conditions aux bords périodiques

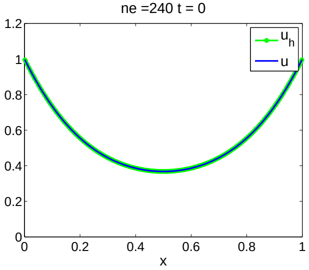


Résultats numériques pour $\Sigma^n_{i,1}$, $\Sigma^n_{i,2}$ et $\Sigma^n_{i,3}$

$$z(x) = sin(\pi x),\ b(x) = x,\ a = 1$$

Conditions aux bords périodiques

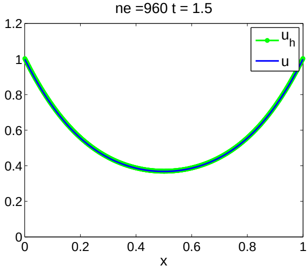


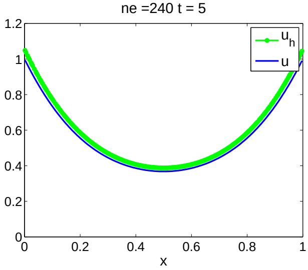


Résultats numériques pour $\Sigma^n_{i,1}$, $\Sigma^n_{i,2}$ et $\Sigma^n_{i,3}$

$$z(x) = sin(\pi x),\ b(x) = x,\ a = 1$$

Conditions aux bords périodiques

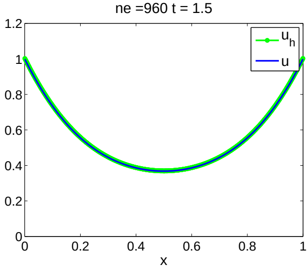


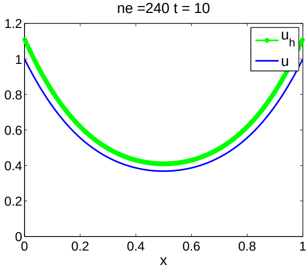


Résultats numériques pour $\Sigma_{i,1}^n$, $\Sigma_{i,2}^n$ et $\Sigma_{i,3}^n$

$$z(x) = sin(\pi x),\ b(x) = x,\ a = 1$$

Conditions aux bords périodiques

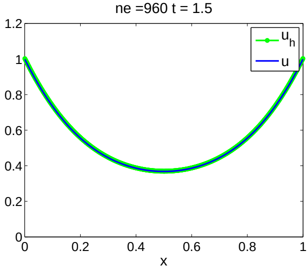


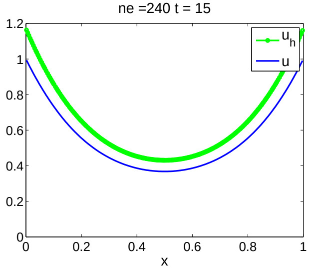


Résultats numériques pour $\Sigma_{i,1}^n$, $\Sigma_{i,2}^n$ et $\Sigma_{i,3}^n$

$$z(x) = sin(\pi x),\ b(x) = x,\ a = 1$$

Conditions aux bords périodiques

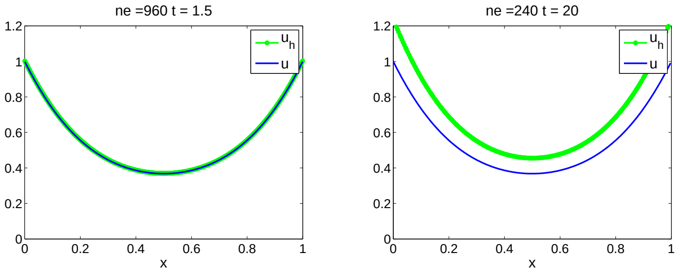




Résultats numériques pour $\Sigma_{i,1}^n$, $\Sigma_{i,2}^n$ et $\Sigma_{i,3}^n$

$$z(x) = sin(\pi x),\ b(x) = x,\ a = 1$$

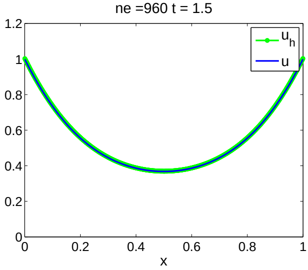
Conditions aux bords périodiques

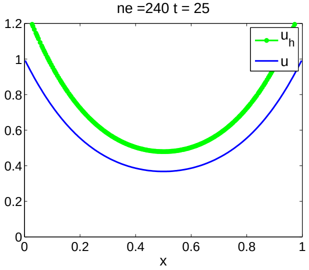


Résultats numériques pour $\Sigma_{i,1}^n$, $\Sigma_{i,2}^n$ et $\Sigma_{i,3}^n$

$$z(x) = sin(\pi x),\ b(x) = x,\ a = 1$$

Conditions aux bords périodiques

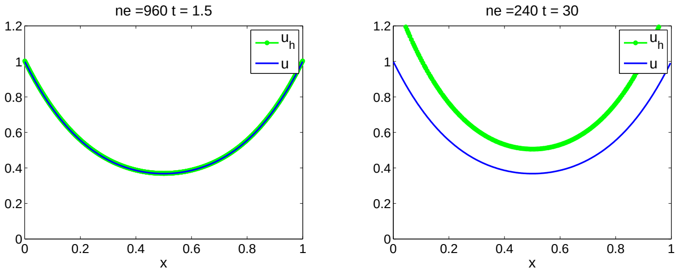




Résultats numériques pour $\Sigma_{i,1}^n$, $\Sigma_{i,2}^n$ et $\Sigma_{i,3}^n$

$$z(x) = sin(\pi x),\ b(x) = x,\ a = 1$$

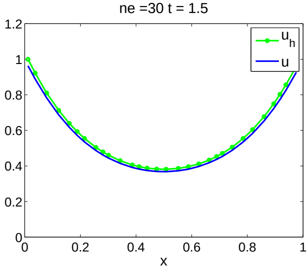
Conditions aux bords périodiques



Nouveau schéma : $\Sigma_{i,4}^{n}$

$$\frac{v_i^{n+1}-v_i^n}{\Delta t} + a\frac{v_i^n - v_{i-1}^n}{\Delta x_i} + \frac{1}{\Delta x_i}\left(\Delta x_{i-1} b(v_{i-1}^n) + \Delta x_i b(v_i^n)\right)\frac{z(x_i)-z(x_{i-1})}{\Delta x_i + \Delta x_{i-1}}$$

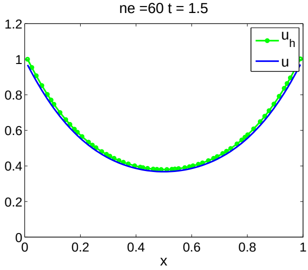
Conditions aux bords périodiques



Nouveau schéma : $\Sigma_{i,4}^{n}$

$$\frac{v_i^{n+1}-v_i^n}{\Delta t} + a\frac{v_i^n-v_{i-1}^n}{\Delta x_i} + \frac{1}{\Delta x_i}\left(\Delta x_{i-1} b(v_{i-1}^n) + \Delta x_i b(v_i^n)\right)\frac{z(x_i)-z(x_{i-1})}{\Delta x_i+\Delta x_{i-1}}$$

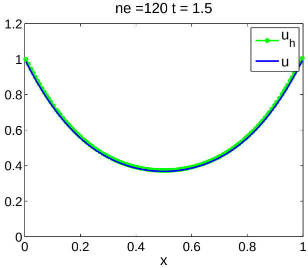
Conditions aux bords périodiques



Nouveau schéma : $\Sigma_{i,4}^{n}$

$$\frac{v_i^{n+1}-v_i^n}{\Delta t}+a\frac{v_i^n-v_{i-1}^n}{\Delta x_i}+\frac{1}{\Delta x_i}\left(\Delta x_{i-1}b(v_{i-1}^n)+\Delta x_i b(v_i^n)\right)\frac{z(x_i)-z(x_{i-1})}{\Delta x_i+\Delta x_{i-1}}$$

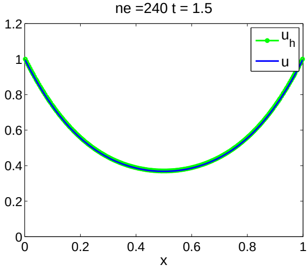
Conditions aux bords périodiques



Nouveau schéma : $\Sigma_{i,4}^n$

$$\frac{v_i^{n+1}-v_i^n}{\Delta t} + a\frac{v_i^n-v_{i-1}^n}{\Delta x_i} + \frac{1}{\Delta x_i}\left(\Delta x_{i-1} b(v_{i-1}^n) + \Delta x_i b(v_i^n)\right)\frac{z(x_i)-z(x_{i-1})}{\Delta x_i+\Delta x_{i-1}}$$

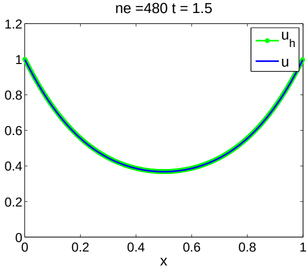
Conditions aux bords périodiques



Nouveau schéma : $\Sigma_{i,4}^{n}$

$$\frac{v_i^{n+1}-v_i^n}{\Delta t} + a\frac{v_i^n-v_{i-1}^n}{\Delta x_i} + \frac{1}{\Delta x_i}\left(\Delta x_{i-1} b(v_{i-1}^n) + \Delta x_i b(v_i^n)\right)\frac{z(x_i)-z(x_{i-1})}{\Delta x_i+\Delta x_{i-1}}$$

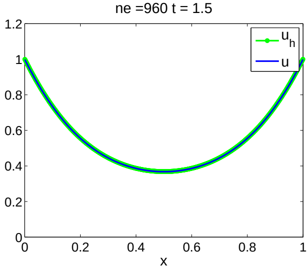
Conditions aux bords périodiques

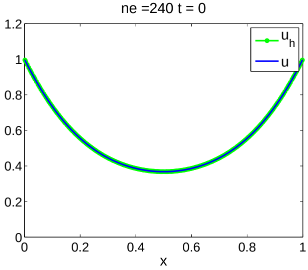


Nouveau schéma : $\Sigma_{i,4}^{n}$

$$\frac{v_i^{n+1}-v_i^n}{\Delta t} + a\frac{v_i^n-v_{i-1}^n}{\Delta x_i} + \frac{1}{\Delta x_i}\left(\Delta x_{i-1} b(v_{i-1}^n) + \Delta x_i b(v_i^n)\right)\frac{z(x_i)-z(x_{i-1})}{\Delta x_i+\Delta x_{i-1}}$$

Conditions aux bords périodiques

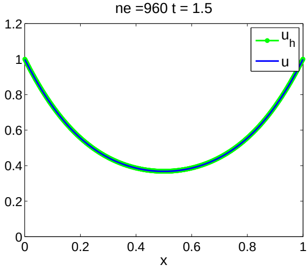


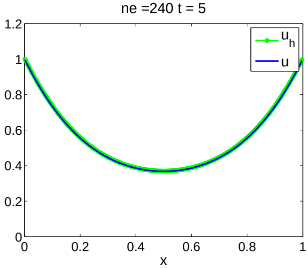


Nouveau schéma : $\Sigma_{i,4}^n$

$$\frac{v_i^{n+1}-v_i^n}{\Delta t} + a\frac{v_i^n-v_{i-1}^n}{\Delta x_i} + \frac{1}{\Delta x_i}\left(\Delta x_{i-1} b(v_{i-1}^n) + \Delta x_i b(v_i^n)\right)\frac{z(x_i)-z(x_{i-1})}{\Delta x_i+\Delta x_{i-1}}$$

Conditions aux bords périodiques

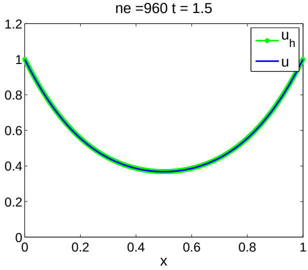


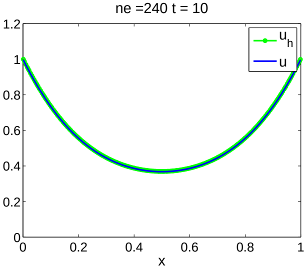


Nouveau schéma : $\Sigma_{i,4}^n$

$$\frac{v_i^{n+1}-v_i^n}{\Delta t} + a\frac{v_i^n-v_{i-1}^n}{\Delta x_i} + \frac{1}{\Delta x_i}\left(\Delta x_{i-1} b(v_{i-1}^n) + \Delta x_i b(v_i^n)\right)\frac{z(x_i)-z(x_{i-1})}{\Delta x_i+\Delta x_{i-1}}$$

Conditions aux bords périodiques

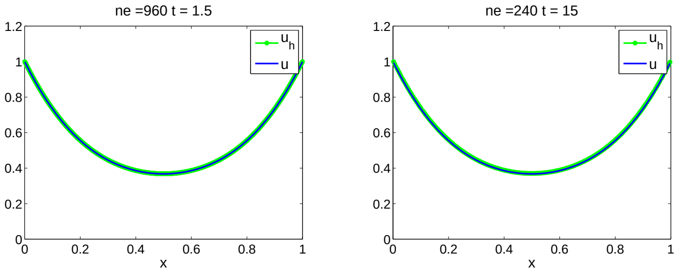




Nouveau schéma : $\Sigma_{i,4}^n$

$$\frac{v_i^{n+1}-v_i^n}{\Delta t} + a\frac{v_i^n - v_{i-1}^n}{\Delta x_i} + \frac{1}{\Delta x_i}\left(\Delta x_{i-1} b(v_{i-1}^n) + \Delta x_i b(v_i^n)\right)\frac{z(x_i)-z(x_{i-1})}{\Delta x_i + \Delta x_{i-1}}$$

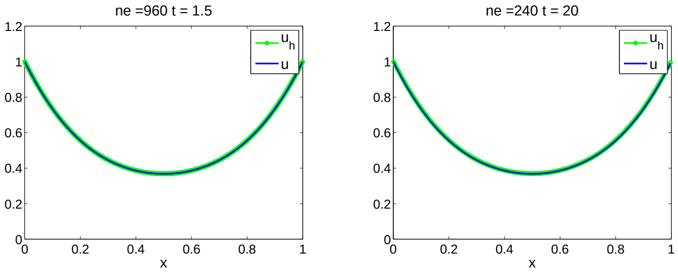
Conditions aux bords périodiques



Nouveau schéma : $\Sigma_{i,4}^{n}$

$$\frac{v_i^{n+1}-v_i^n}{\Delta t} + a\frac{v_i^n-v_{i-1}^n}{\Delta x_i} + \frac{1}{\Delta x_i}\left(\Delta x_{i-1} b(v_{i-1}^n) + \Delta x_i b(v_i^n)\right)\frac{z(x_i)-z(x_{i-1})}{\Delta x_i+\Delta x_{i-1}}$$

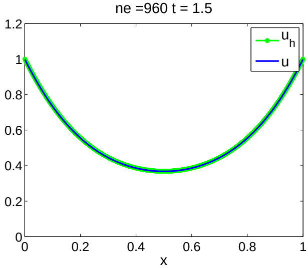
Conditions aux bords périodiques

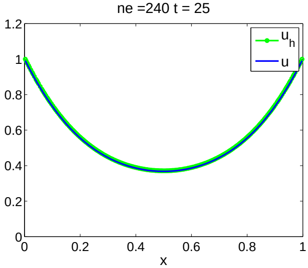


Nouveau schéma : $\Sigma_{i,4}^{n}$

$$\frac{v_i^{n+1}-v_i^n}{\Delta t} + a\frac{v_i^n-v_{i-1}^n}{\Delta x_i} + \frac{1}{\Delta x_i}\left(\Delta x_{i-1} b(v_{i-1}^n) + \Delta x_i b(v_i^n)\right)\frac{z(x_i)-z(x_{i-1})}{\Delta x_i+\Delta x_{i-1}}$$

Conditions aux bords périodiques

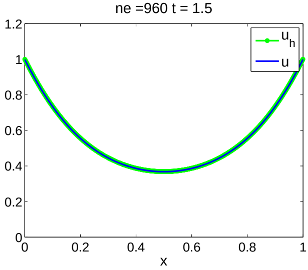


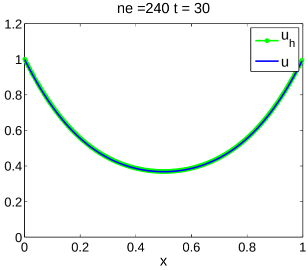


Nouveau schéma : $\Sigma_{i,4}^n$

$$\frac{v_i^{n+1}-v_i^n}{\Delta t} + a\frac{v_i^n - v_{i-1}^n}{\Delta x_i} + \frac{1}{\Delta x_i}\left(\Delta x_{i-1} b(v_{i-1}^n) + \Delta x_i b(v_i^n)\right)\frac{z(x_i)-z(x_{i-1})}{\Delta x_i + \Delta x_{i-1}}$$

Conditions aux bords périodiques





Equation des ondes en 2D

$$\frac{\partial U}{\partial t} + \underbrace{\begin{pmatrix} 0 & 1 & 0 \\ c^2 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}}_{A_1} \frac{\partial U}{\partial x} + \underbrace{\begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ c^2 & 0 & 0 \end{pmatrix}}_{A_2} \frac{\partial U}{\partial y} = 0.$$

$$F(U) = (A_1 U, A_2 U)$$

$$\Phi_{K,L} = |K \cap L| \left(\frac{F(U_K^n) + F(U_L^n)}{2} \odot n_{K,L} - sign(\underbrace{F \odot n_{K,L}}_{A_{K,L}}) \frac{F(U_L^n) - F(U_K^n)}{2} \odot n_{K,L} \right)$$

$$\frac{U_K^{n+1} - U_K^n}{\Delta t} + \Phi_K = 0.$$

Schéma obtenu par la méthode VF

$$\frac{U_K^{n+1} - U_K^n}{\Delta t} + \sum_{L \in N(K)} (A_{K,L}^+ U_K^n + A_{K,L}^- U_L^n) = 0$$

$$A_{K,L}^+ = |K \cap L| \frac{A_{K,L} + sign(A_{K,L}) A_{K,L}}{2} \quad A_{K,L}^- = |K \cap L| \frac{A_{K,L} - sign(A_{K,L}) A_{K,L}}{2}$$

$$\sum_{L \in \mathcal{N}(K)} A_{K,L}^+ (\Gamma_K + G_K - G_{K,L}) + \sum_{L \in \mathcal{N}_0(K)} A_{K,L}^- (\Gamma_L + G_L - G_{K,L}) = 0$$

Similitudes avec l'équation de transport linéaire
Le système s'écrit $A\Gamma = \Delta$.
Question : A inversible ?

Une conjecture sur le nombre de valeurs propres nulles de A

Conjecture

$Dim(Ker(A)) = \frac{T - A_b}{2} + 1$

Proposition

$\frac{T - A_b}{2} + 1 = S_{int}$

Preuve.

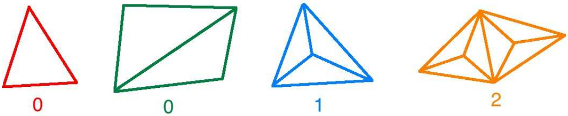
Euler : $T - A + S = 1$ puis $2T - 2A + 2S = 2$

$2A_{int} + A_b = 3T \Leftrightarrow 2A - A_b = 3T$

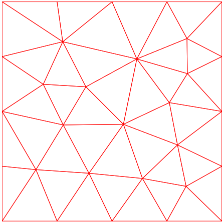
On somme : $2T - A_b + 2S = 2 + 3T$

Soit $2(S_{int} + A_b) = T + A_b + 2$ (car $A_b = S_b$)

Vérifications



Exemples simples



Exemple Matlab : T $= 42$, $A_b = 16$, $S_{int} = 14$

Bibliographie

[1] Laurent Di Menza. Analyse numérique des équations aux dérivés partielles. Enseignement des Mathématiques (Cassini) 24. Paris : Cassini. xii, 221 p., 2009.

[2] Brigitte Lucquin. Equations aux dérivées partielles et leurs approximations. Niveau M1. Paris : Ellipses. v, 227 p., 2004.

[3] Frédéric Pascal. On supra-convergence of the finite volume method for the linear advection problem. In Paris-Sud Working Group on Modelling and Scientific Computing 2006–2007, volume 18 of ESAIM Proc., pages 38–47. EDP Sci., Les Ulis, 2007.

[4] Daniel Bouche, Jean-Michel Ghidaglia, and Frédéric Pascal. Error estimate and the geometric corrector for the upwind finite volume method applied to the linear advection equation. SIAM J. Numer. Anal., 43(2) :578–603 (electronic), 2005.

[5] Chiara Simeoni. Remarks on the consistency of upwind source at interface schemes on nonuniform grids. J. Sci. Comput., 48(1-3) :333–338, 2011.

Le Transport Optimal

Théorie générale et optimisation de la continuité

Deleu T., Ducasse R., Sacchelli L.

ENSC

Mardi 03 Juillet 2012

Le Transport Optimal

Le Transport Optimal

Notions de base

Mesure image et transport

Soient (Ω, F), (Ω', F') deux espaces mesurables, μ, ν deux mesures sur ces espaces. On définit la mesure image $T_{\#}\mu : A \in F' \longmapsto \mu(T^{-1}(A))$.
On dit que T est un transport entre μ et ν si $T_{\#}\mu = \nu$ (on note $T(\mu, \nu)$ l'ensemble des transports entre μ et ν).

Notions de base

Mesure image et transport

Soient (Ω, F), (Ω', F') deux espaces mesurables, μ, ν deux mesures sur ces espaces. On définit la mesure image $T_{\#}\mu : A \in F' \longmapsto \mu(T^{-1}(A))$.
On dit que T est un transport entre μ et ν si $T_{\#}\mu = \nu$ (on note $T(\mu, \nu)$ l'ensemble des transports entre μ et ν).

Problème de Monge

Étant donné une fonction coût $c : (\Omega \times \Omega') \longrightarrow \bar{\mathbb{R}}$, on définit le transport optimal comme étant l'élément de $T(\mu, \nu)$ minimisant la fonctionnelle :

$$T \in T(\mu, \nu) \longmapsto \int_{\Omega} c(x, T(x)) d\mu(x)$$

Le Transport Optimal

Problématique

S'inspirant du problème de Monge, on veut minimiser une autre fonctionnelle : l'oscillation à l'échelle δ du transport T.

$$F_\delta : T \in T(\mu, \nu) \longmapsto \sup_{\|x-y\| \leq \delta} \|T(x) - T(y)\|$$

Problématique

S'inspirant du problème de Monge, on veut minimiser une autre fonctionnelle : l'oscillation à l'échelle δ du transport T.

$$F_\delta : T \in T(\mu, \nu) \longmapsto \sup_{\|x-y\| \leq \delta} \|T(x) - T(y)\|$$

- Origine du problème : cryptage de données de géolocalisation de téléphones portables
- But : Approcher numériquement ce transport optimal

D'anciens résultats

À l'aide d'une démonstration technique, le groupe de stage de l'année dernière à obtenu le résultat suivant :

Théorème

Soient Ω un compact de $\mathbb{R}$, μ et ν des mesures de probabilités sur Ω. Si μ est sans atome, alors il existe un transport optimal $T \in T(\mu, \nu)$ pour le coût F_δ où

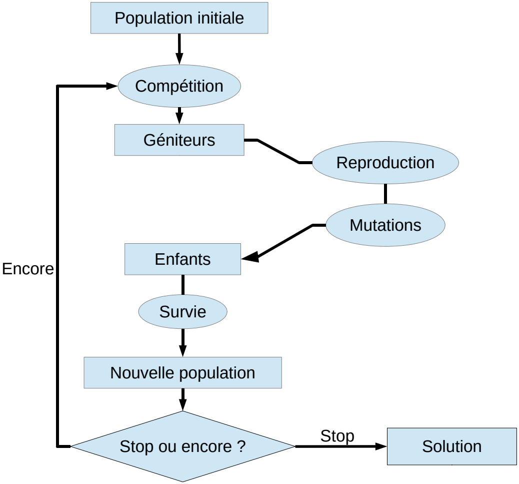
$$F_\delta : T \longmapsto \sup_{\|x-y\| \leq \delta} \|T(x) - T(y)\|$$

Preuve fondée sur la construction d'un transport optimal. Quelques propriétés de l'optimum :

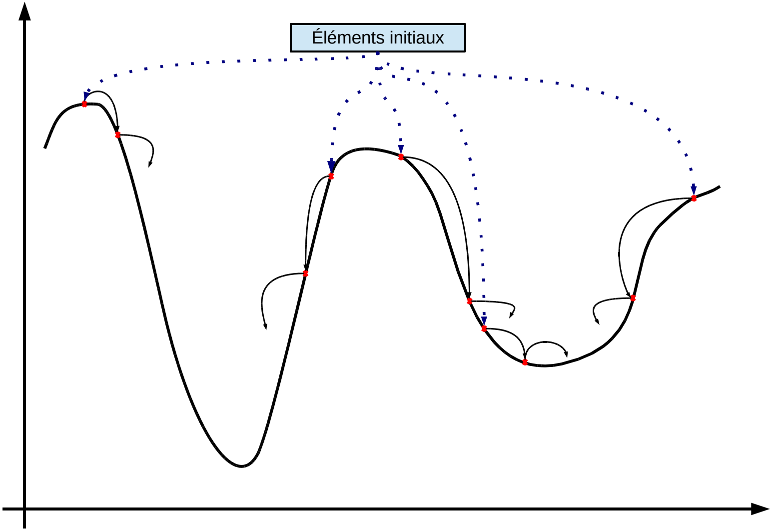
- Monotone par morceaux
- Intervalles de monotonie de longueur $\geq 2\delta$

Algorithmes génétiques

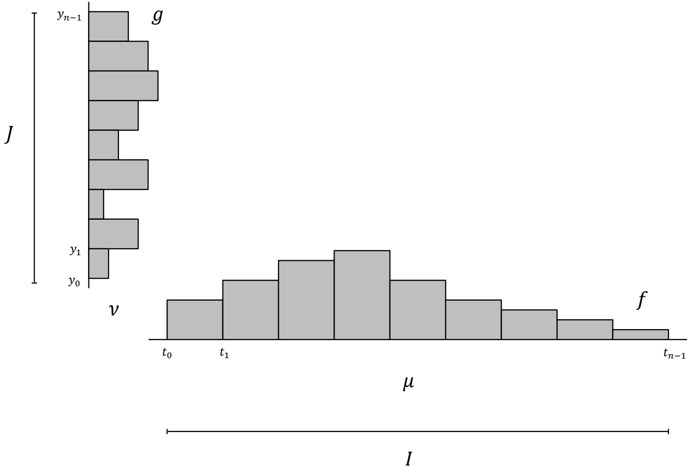
But : diminution de F_δ au cours de l'évolution



Algorithmes génétiques



Notations et représentation



Individus

Le transport optimal est monotone par morceaux.
Chaque morceau correspond au transport de μ sur une partie de ν.
On représente donc les individus par la décomposition de ν en $< \nu_0, \nu_1, \ldots, \nu_{m-1} >$ tel que

$$\nu_0 + \nu_1 + \ldots + \nu_{m-1} = \nu$$

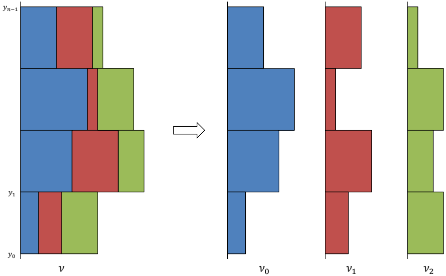
Initialisation

Création d'un individu $< \nu_0, \nu_1, \ldots, \nu_{m-1} >$ aléatoire :

- On tire aléatoirement la masse (totale) qu'on veut alouer à ν_j, notée α_j.
- On découpe chaque morceaux de la densité de ν suivant une loi uniforme élevée à la puissance p_j telle que

$$\mathbb{E}(X^{p_j}) = \alpha_j \qquad \Longleftrightarrow \qquad p_j = \frac{1}{\alpha_j} - 1$$

- On crée la mesure ν_j avec la densité ainsi créée

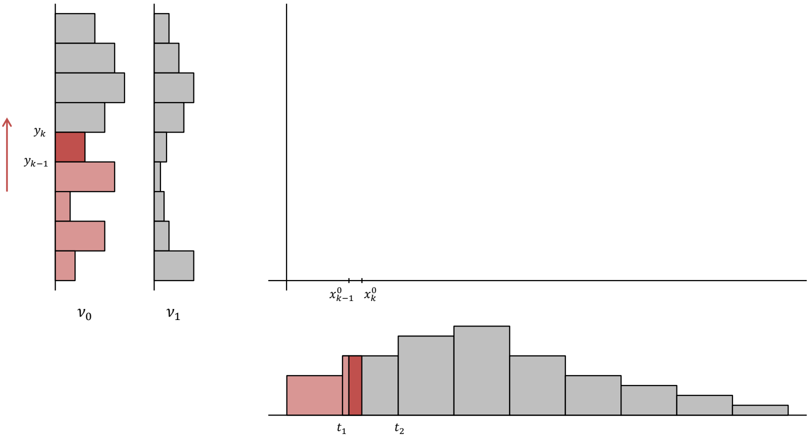


Calcul de F_δ

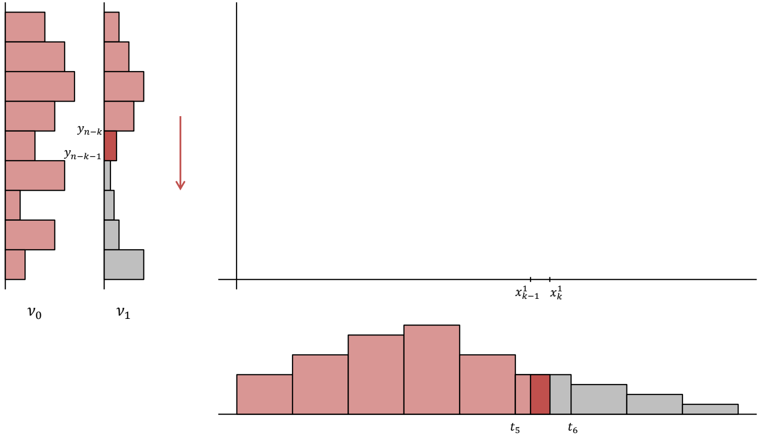
La discrétisation (t_k) n'est pas adaptée au calcul de $F_\delta(T)$.
On crée une nouvelle discrétisation (x_k^j) de l'espace de départ.

$$\int_{t_0}^{x_k^j} \mathrm{d}\mu = \int_J \mathrm{d}(\nu_0 + \ldots + \nu_{j-1}) \; + \; \int_{y_0}^{y_k} \mathrm{d}\nu_j \qquad \text{si } j \bmod 2 = 0$$

$$\int_{t_0}^{x_k^j} \mathrm{d}\mu = \int_J \mathrm{d}(\nu_0 + \ldots + \nu_{j-1}) \; + \; \int_{y_{n-1-k}}^{y_{n-1}} \mathrm{d}\nu_j \qquad \text{si } j \bmod 2 = 1$$



$$\int_{t_0}^{x_k^j} \mathrm{d}\mu = \int_J \mathrm{d}(\nu_0 + \ldots + \nu_{j-1}) + \int_{y_0}^{y_k} \mathrm{d}\nu_j \qquad \text{si } j \bmod 2 = 0$$



$$\int_{t_0}^{x_k^j} \mathrm{d}\mu = \int_J \mathrm{d}(\nu_0 + \ldots + \nu_{j-1}) \; + \int_{y_{n-1-k}}^{y_{n-1}} \mathrm{d}\nu_j \qquad \text{si } j \bmod 2 = 1$$

Calcul de F_δ

Hypothèse : Comme T optimal est monotone par morceaux, on calcule F_δ sur les morceaux où T est monotone.
On a alors, pour x, y dans le même intervalle de monotonie :

$$F_\delta(T) = \sup_{|x-y|\leq\delta} |T(x) - T(y)| = \sup_{|x-y|=\delta} |T(x) - T(y)|$$

On utilise la discrétisation (x_k^j) car :

$$|T(x_k^j) - T(x_l^j)| = |y_k - y_l| = \mathrm{pas}_\nu \times |k - l|$$

Pour calculer $F_\delta(T)$, il suffit de trouver x_k^j et x_l^j tels que

$$|x_k^j - x_l^j| \approx \delta$$

Algorithme dynamique ayant une complexité temporelle en $O(n \times m)$.

Reproduction

Reproduction simple : à partir de 2 individus parent_1 et parent_2, on génère :

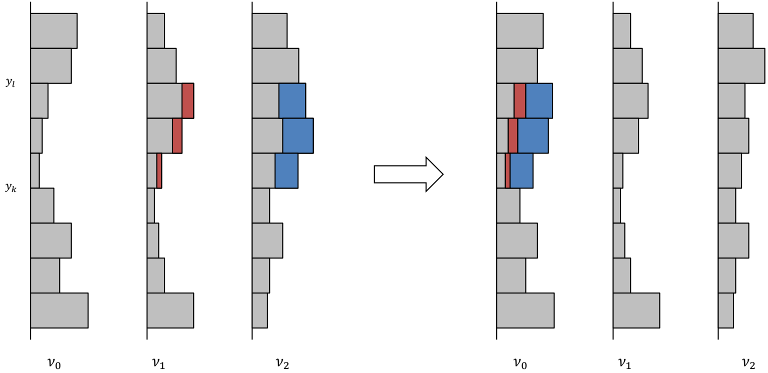
$$\mathrm{enfant}_1 = \frac{2}{3} \times \mathrm{parent}_1 + \frac{1}{3} \times \mathrm{parent}_2$$

$$\mathrm{enfant}_2 = \frac{1}{3} \times \mathrm{parent}_1 + \frac{2}{3} \times \mathrm{parent}_2$$

Mutation

Mutation dite par *compensation de masse* basée sur 2 opérations :

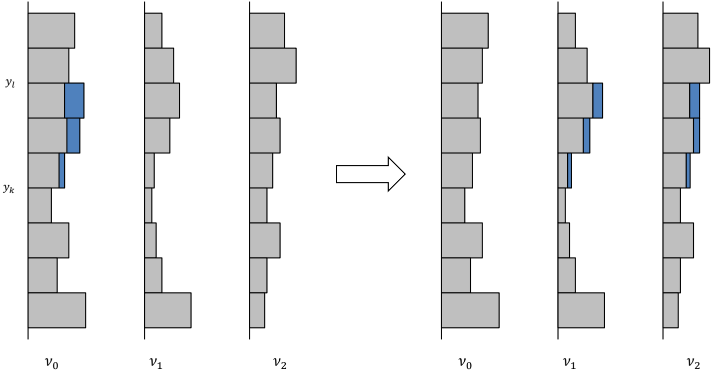
- **Mutation par diminution de pente**
  Là où la pente de T est *forte*, cela signifie que la masse correspondante dans l'individu est trop *faible*. L'idée est d'y ajouter de la masse



Mutation

Mutation dite par *compensation de masse* basée sur 2 opérations :

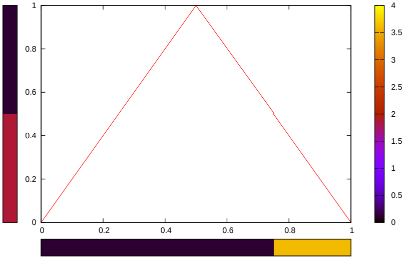
- **Mutation par augmentation de pente**
  Là où la pente de T est *faible*, cela signifie que la masse correspondante dans l'individu est trop *forte*. L'idée est d'y enlever de la masse

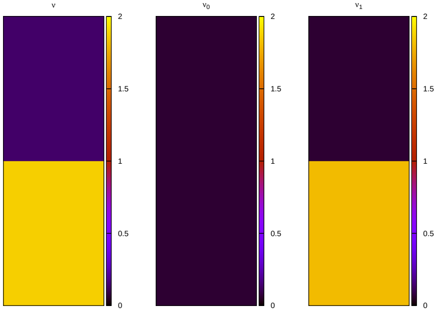


Attention : va à l'encontre de notre but. Opération que l'on effectue rarement (15% dans la pratique).

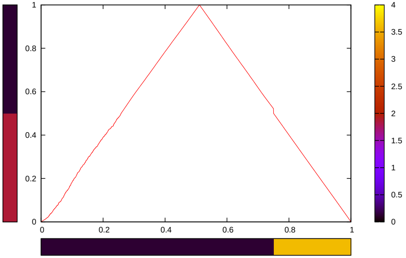
Exemple

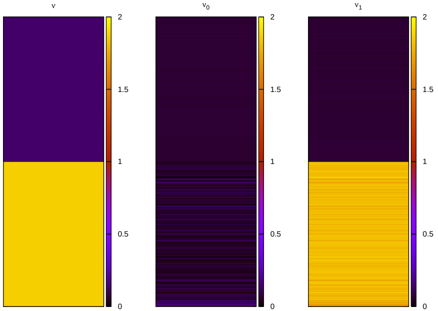
Dans cet exemple, on construit μ et ν de la manière suivante





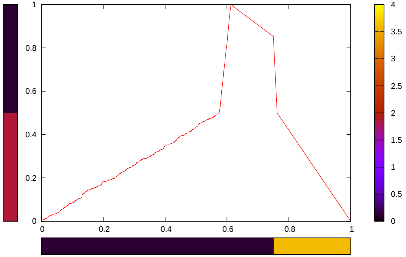
Résultats

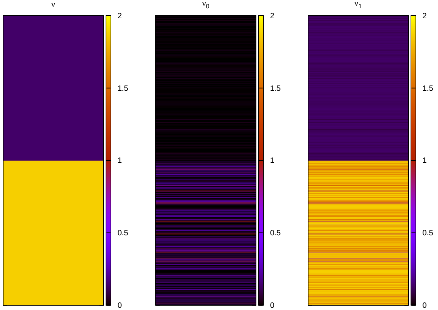




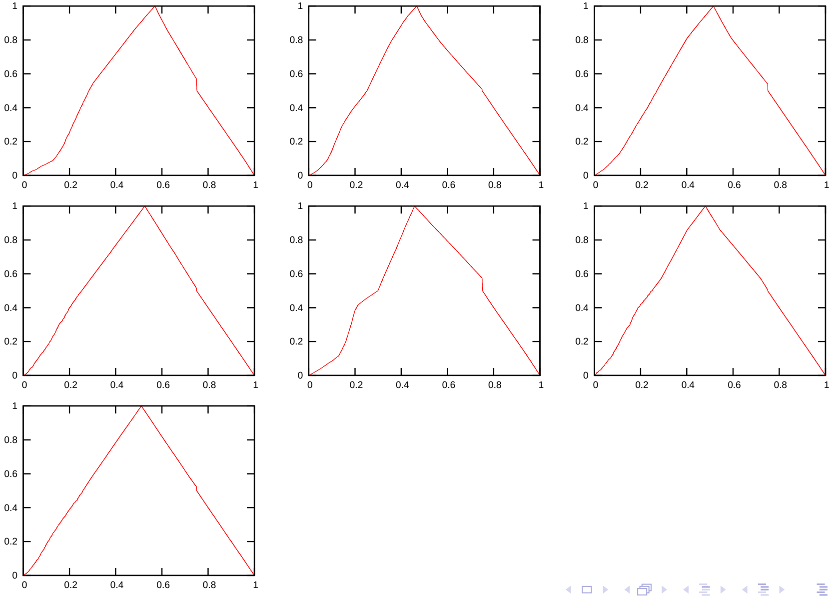
Exemple

Autre minimum local

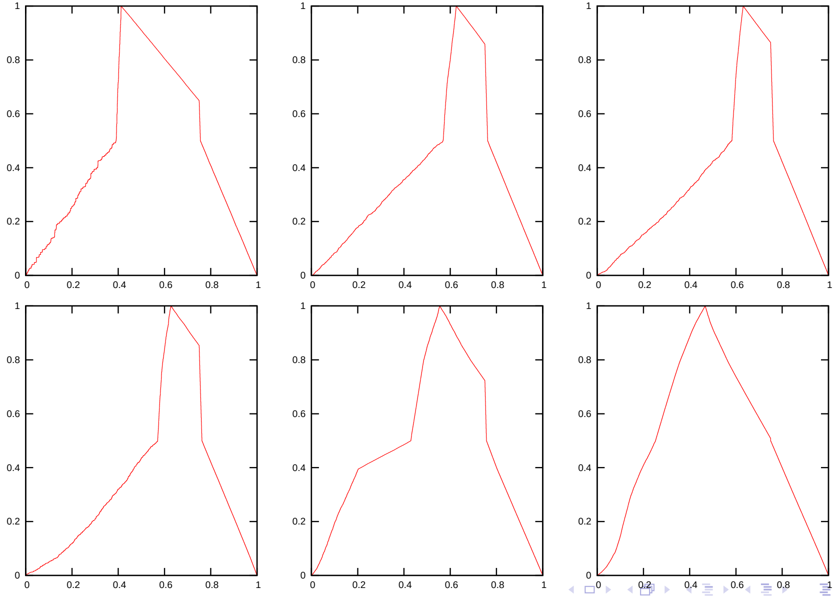




Exemple



Exemple



Panoramas

Soutenance de stage

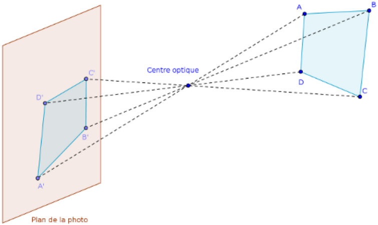
Julie Gauthier, Samy Jaziri

École Normale Supérieure de Cachan

Mardi 3 juillet 2012

Position du problème

La géométrie projective : un point de vue adapté au problème



Point de vue	euclidien	projectif
Notation	$\mathbb{R}^2$	$\mathbb{P}^2$
Coordonnées d'un point	(x, y)	$\alpha(x, y, 1)$
Coordonnées d'une droite	$ax + by + c = 0$	(a, b, c)
Les droites parallèles...	ne se coupent pas.	se coupent !

Les points de la géométrie projective représentent les droites de $\mathbb{R}^3$.

Une transformation qui conserve l'alignement

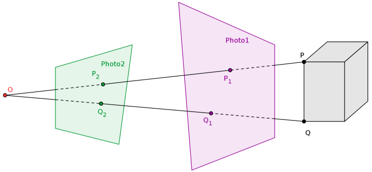
Définition

Les **transformations projectives** ou **homographies** sont les transformations inversibles qui conservent l'alignement.

Théorème

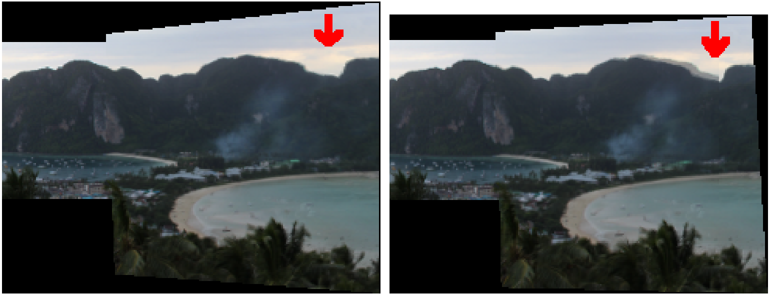
Une application $h : \mathbb{P}^2 \to \mathbb{P}^2$ *est une homographie ssi il existe* $H \in GL_3(\mathbb{R})$ *telle que* $\forall x \in \mathbb{P}^2, h(x) = Hx$.

H est une matrice **homogène** : ses coordonnées sont données à un coefficient multiplicatif près $\implies$ 4 points.



SIFT et RANSAC

SIFT trouve les points particuliers de chaque photo et les apparie. RANSAC trouve une homographie qui fait correspondre les paires de points particuliers.

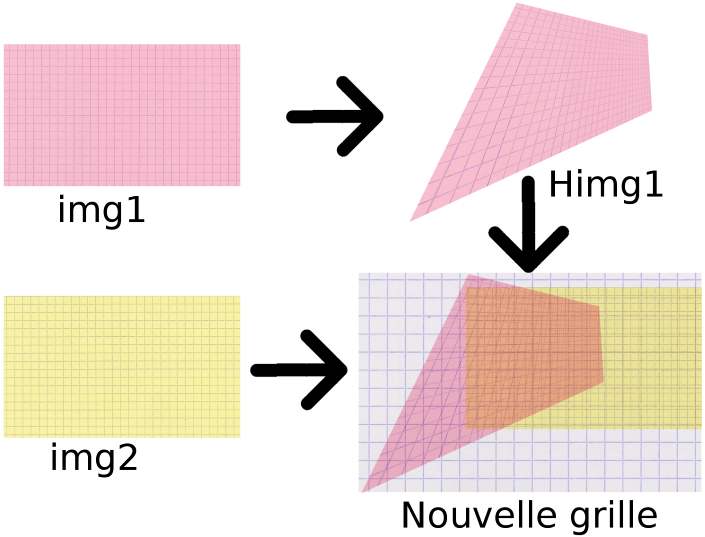


Notre algorithme

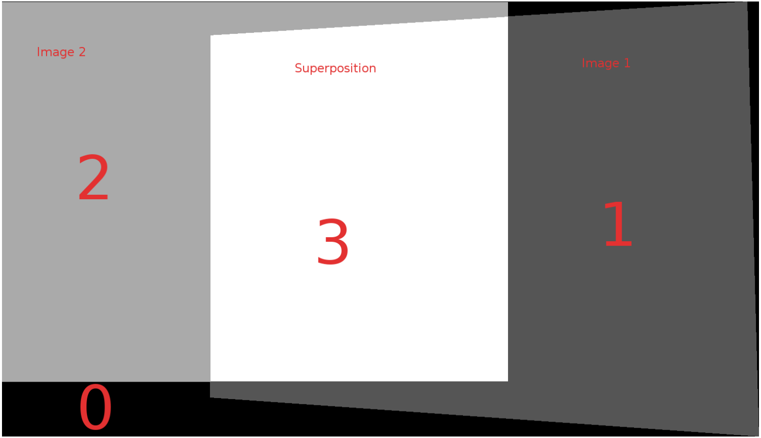
En entrée : deux images `img1` et `img2`.
En sortie : une image, le panorama `pan` obtenu en appliquant la bonne homographie à `img1` pour qu'elle se superpose sur l'image `img2`.
Algorithme :

1. trouver et apparier les points particuliers de `img1` et `img2` (SIFT),
2. trouver une homographie `H` qui convient (RANSAC),
3. appliquer `H` à `img1`, on appelle `Himg1` le résultat,
4. superposer `Himg1` et `img2`, on appelle `pan` le résultat,
5. renvoyer `pan`.

Ré-échantillonage : étape 3

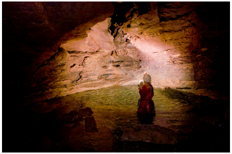


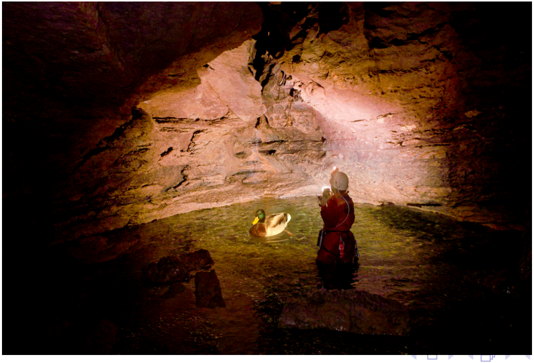
Collage : étape 4



Collage en faisant la moyenne

Utilisation des équations de Poisson : incrustation





Application au panorama

(i) Photo 1

(j) Photo 2

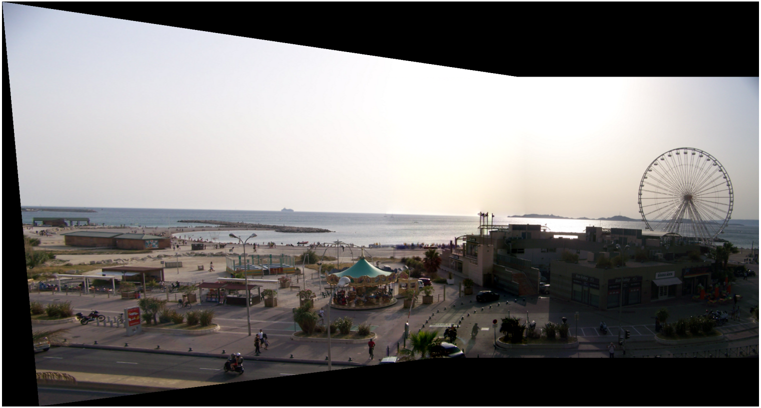
(k) 1

(l) 2

(m) moyenne

(n) max

Conclusion



Bibliographie

Recognising panoramas Brown et Lowe, 2003

Automatic image stitching using invariant features Brown et Lowe, 2007

Multiple view geometry in computer vision Hartley et Zisserman, 2003

Object recognition from local scale-invariant features Lowe, 1999

vlfeat.org Vedaldi et Fulkerson, 2008

Ransac algorithm with example of finding homography Wiggin, 2011

Random sample consensus Fischler et Bolles, 1981

Poisson image editing Pérez, Gangnet et Blake, 2003

Ko Phi Phi, Thaïlande

Modèles continus de supply chains. Application à la modélisation d'une architecture de calcul parallèle.

B. Dadoun L. Feuilloley S. Zhang

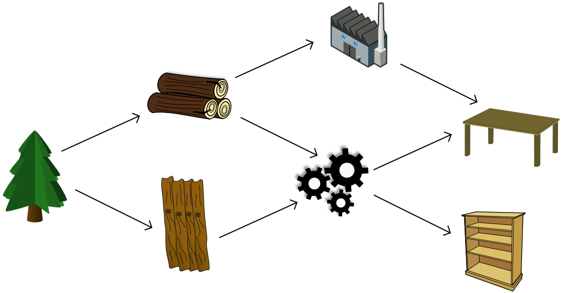
Équipe encadrante : F. De Vuyst, F. Salvarani, D. Bouche, P. Jaisson

3 juillet 2012

Sommaire

Qu'est-ce qu'une supply chain ?

- Terme français : « chaîne de distribution »
- Une dynamique : des flux d'entrée, de sortie ; des temps d'attente, de traitement



Objectifs

- Étudier les systèmes de lois de conservation hyperboliques dans le contexte des supply chains ;
- Reprendre des modèles numériques de supply chains ;
- Adapter ces modèles à la simulation d'une architecture de calcul parallèle.

Documents étudiés

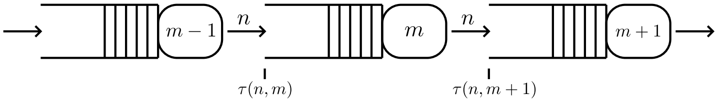
C. D'Apice, S. Göttlich, M. Herty, B. Piccoli
Modeling, Simulation and Optimization of Supply Chains : A continuous approach, SIAM, 2010.

S. Williams
Auto-tuning Performance on Multicore Computers, thèse à l'Université de Berkeley, 2008.

Sommaire

Processeurs et files d'attente

- Fournisseurs (= processeurs) $m = 1, \ldots, M$, objets (matière) n
  Temps de traitement $T(m)$, capacité $\mu(n, m)$



- $\tau(n, m+1)$: instant où l'objet n sort de m (= arrive à $m+1$).

$$\boxed{\tau(n,m+1) = \max\left\{\tau(n,m) + T(m), \tau(n-1,m+1) + \frac{1}{\mu(n-1,m)}\right\}.}$$

Loi de conservation discrète

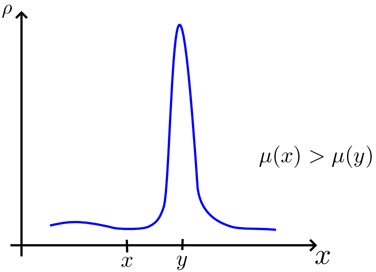
- Exprimer le flux f en fonction de la densité de matière ρ :
  - vitesse de traitement $\times$ densité de matière ;
  - restreint par la capacité du processeur.

Théorème

$$f(\tau(n,m),x_m) = \min\left\{\mu(n,m-1), \frac{h_{m-1}\rho(\tau(n,m),x_{m-1})}{T(m-1)}\right\}.$$

Loi de conservation continue ($M \to +\infty$)

- Formulation faible car congestionnement possible



Théorème

Si les temps de traitement et de passage des objets sont bornés, alors :

$$\partial_t \rho + \partial_x f = 0, \quad f = \min\{\mu, \rho\}.$$

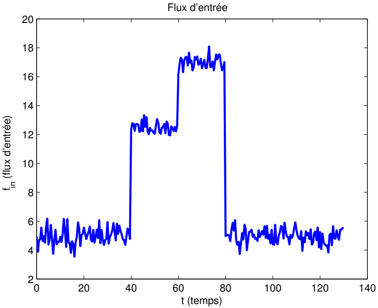
Dans la réalité

- Nombre fini de processeurs... Théorème non applicable ?
- Ruse. Augmenter virtuellement le nombre de processeurs :
  - démultiplier chaque processeur m en K sous-processeurs « virtuels » ;
  - diviser les capacités des processeurs par K.

Simulation numérique

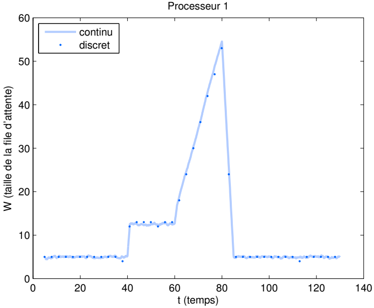
Comparaison des modèles discret et continu

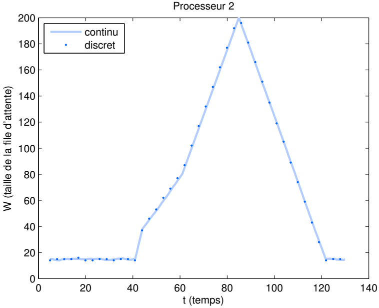
- Modèle discret : adapté à des problèmes de petite taille ;
- Modèle continu : adapté aux problèmes de plus grande taille.



Simulation numérique (suite)
Comparaison des modèles discret et continu

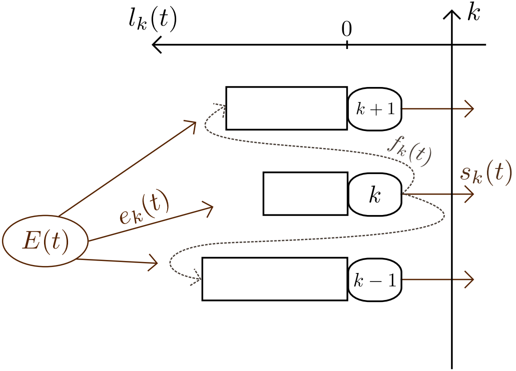
- État des files d'attente en fonction du temps :





Modélisation d'unités en parallèle (MUP)

- Unités k, de longueur l_k.
- Flux externes : $e_k(t)$ (entrée), $s_k(t)$ (sortie), internes : $f_k(t)$.



Équations thermodynamiques

- Diffusion + convection :

$$f_k = -\lambda(I_{k+1} - I_k)$$
$$s_k = hI_k$$

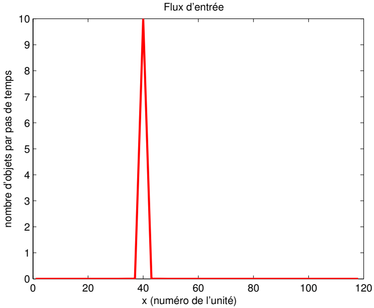
- Équation :

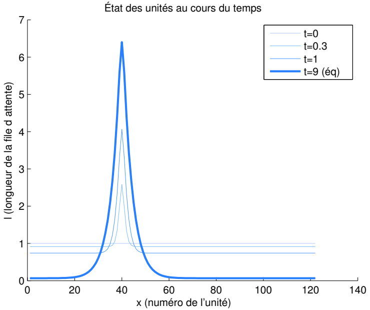
$$\frac{\partial I(x,t)}{\partial t} = \lambda \frac{\partial^2 I(x,t)}{\partial x^2} + e(x,t) - hI(x,t)$$

- Schéma :

$$I_j^{n+1} = I_j^n + \frac{\lambda \Delta t}{2\Delta x}\left(I_{j+1}^n - 2I_j^n + I_{j-1}^n\right) + \Delta t\left(e_j^n - hI_j^n\right)$$

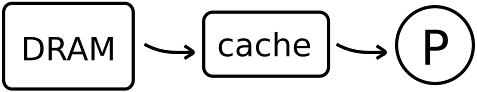
Simulation numérique



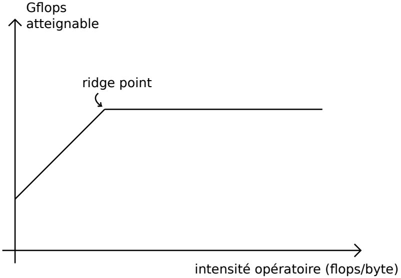


Sommaire

Modèle Roofline



$$\text{Gflops} = \min\{\text{Gflops(crête)}, \text{bande passante} \times \text{intensité opératoire}\}$$

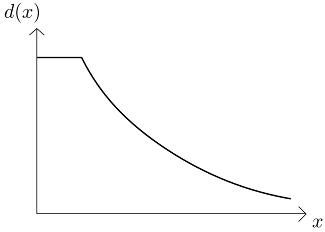


Vitesse en fonction du trafic

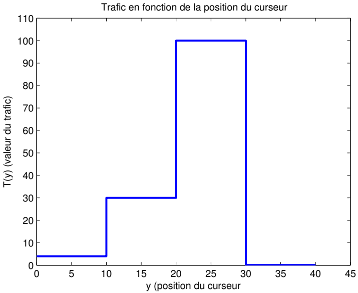
- $T(y)$: trafic nécessaire au point y (position du curseur) dans le programme
- Vitesse d'exécution y' décroissante avec le trafic :

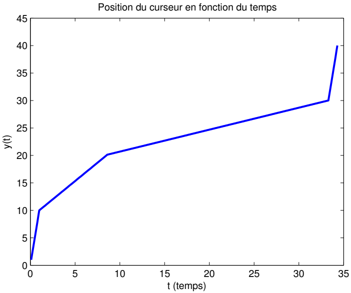
$$y'(t) = d\,(T(y(t)))\,,$$

avec d :



Résultats


Trafic en fonction de la position du curseur
T(y) (valeur du trafic)
y (position du curseur
0
10
20
30
40
50
60
70
80
90
100
110
0
5
10
15
20
25
30
35
40
45


Position du curseur en fonction du temps
y(t)
t (temps)
0
5
10
15
20
25
30
35
40
45
0
5
10
15
20
25
30
35

Conclusion

- Étude bibliographique ;
- Approche files d'attente dans la modélisation ;
- Échec de l'approche pour la modélisation d'une architecture de calcul parallèle.
- Approche plus réaliste à partir du modèle Roofline.

Jusqu'où ne pas aller trop loin avec les moyennes de Fréchet ?

Fauchier-Magnan Antoine, Chevallier Augustin, Richard Kevin

ENS Cachan

2 juillet 2012

- **Reconstituer un signal représentatif d'une population**.
- L'observation Y donne une information sur s bruitée et modulo une translation non observable (variable cachée).

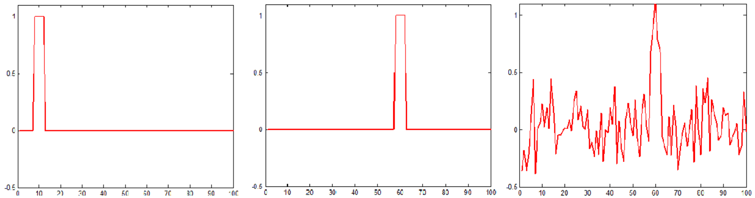


FIGURE: Signal initial, translaté, translaté et bruité.

Observables

σ connu, on définit les observables par : $Y_i = \tau_i.s + \sigma_i, \quad i \in [\![1, J]\!]$

- τ_i loi uniforme
- $\sigma_i \sim N(0, \epsilon^2 I_N)$

- Si on connaissait τ, on pourrait définir l'estimateur par $\widehat{s} = \frac{1}{J}\sum_{j=1}^{J} \tau_j^{-1} Y_j$ et $\lim_{J\to\infty} \widehat{s} = s$
- **Question** : τ_j étant aussi aléatoire, comment estimer s ?

Enjeux

- Ici, on se concentre sur un modèle jouet.
- Permet de comprendre le problème de l'estimation du point de vue des moyennes de Fréchet dans les espaces quotients.

- **MaxMax** : Approche géométrique qui a une interprétation statistique (ou vice versa).

Algorithme "MaxMax"

Principe : Chercher l'estimateur du maximum de vraisemblance :
$\widehat{s,\tau} = argmax_{s,\tau}\, q(Y, s, \tau) = argmin_{s,\tau_1,\ldots,\tau_N} \sum_i \|\tau_i.y_i - s\|_2^2$

- On utilise un algorithme itératif pour minimiser sur s, τ : on calcule le max sur τ, puis avec les translations obtenues, sur s. Et on recommence.

- **observation** : Minimiser sur s, à τ_i fixés, c'est faire la moyenne empirique. Ici, on fait une moyenne sur les osbervations modulo les translations. On travaille alors sur un espace quotient, et minimiser s correspond à faire la moyenne de Fréchet.

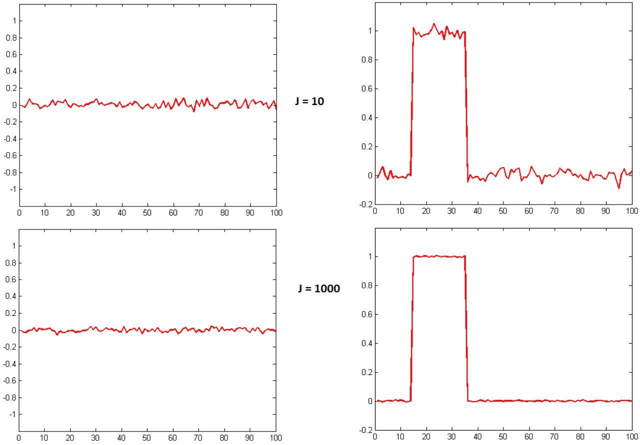
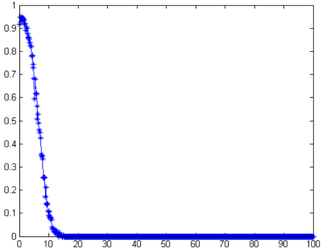


FIGURE: Comparaison des résultats obtenus par l'algorithme pour un signal constant et un créneau ($\sigma = 0.1$)

- Convergence de l'algorithme semble dépendre du signal de départ.
- Difference entre un crénau et une constante : le nombre de translations correctement estimées.
- On introduit donc $f(s) = min_{\tau \neq 0} ||s - \tau.s||$

FIGURE: Probabilité d'erreur en fonction de $f(s)$



- **Vision géométrique du problème**

Moyenne de Frechet

Soit M une variété Riemannienne. On définit x la moyenne de Fréchet de $x_1, ..., x_n \in M$ comme suit :
$x = argmin_{y \in M} \sum_i d_M(y, x_i)^2$

- **Remarque** La moyenne de Fréchet dans un espace vectoriel correspond à la moyenne arithmétique ;

$[M] = M/G$

Soit G un groupe de Lie. Supposons la distance sur M équivariente, ie $d_M(g.x, g.y) = d_M(x, y)$.
On pose [M] =M/G.
On a $d_{[M]}(x, y) = inf_{g,g'} d_M(g.x, g'.y) = inf_g d_M(x, g.y)$.

Moyenne de Frechet dans$[M] = M/G$

La moyenne de Frechet de $m_1, ..., m_n \in M$ dans M/G est

$$\begin{aligned} argmin_{[x]} \sum_i d_{M/G}([x],[x_i])^2 &= argmin_x \sum_i inf_g d_M(x, g.x_i)^2 \\ &= argmin_{x,g_1,\ldots,g_n} \sum_i d_M(x, g_i.x_i)^2 \end{aligned}$$

- Remarque : on n'a pas de théoreme d'existence ou d'unicité de la moyenne de Frechet.

Lien avec l'algorithme

M est l'espace des signaux et G celui des transformations. On a alors $y_1, .., y_J \in M, \tau_1, ..., \tau_J \in G$
Chercher $argmin_{s,\tau_1,\ldots,\tau_N} \sum_i \|\tau_i.y_i - s\|_2^2$ revient à prendre la moyenne de Frechet dans $[M] = M/G$ des $[y_1], .., [y_J]$.

Modélisation

Pour modéliser $J \to \infty$, plutôt que d'utiliser des $y_1, ..., y_J$ on prend ν une densité de probabilité sur M.

- Points fixes pour G : constantes.
- **Contribution au stage** : théorème montrant la non convergence de l'algorithme pour les constantes

Théorème

Soit $m_0 \in M$ un point fixe pour G.
Si M est une variété riemannienne complète, ν charge tout ouvert, $\int_M d_M(m,z)^2 d\nu(z) < \infty$ et $M \setminus Fix(G)$ dense dans M,
alors m_0 ne peut pas être une moyenne de Frechet pour ν.

- Algorithme itératif visant à retrouver un représentant d'une classe dans un modèle à données manquantes.
- Itération à l'étape t :
  - Calculer l'espérance :
    $$\mathbb{E}[\log q(Y, \tau_j, s) | Y, s_t] = -\frac{1}{2\epsilon^2} \sum_{j=1}^{J} \sum_{k} \| Y_j - \tau_k . s \|^2 \mathbb{P}(\tau_j = \tau_k | Y_j, s_t)$$
  - Maximiser sur s :
    $$s_{t+1} = \frac{1}{J} \sum_{j=1}^{J} \sum_{k} \tau_k^{-1} . Y_j \, \mathbb{P}(\tau_j = \tau_k | Y_j, s_t)$$

Principe du SAEM

Version plus rapide de l'algorithme EM : utilisation d'une approximation stochastique.

- Statistique exhaustive : $A(Y_j, s) = \tau_j^{-1}.Y_j$
- Itération à l'instant t :
  - τ_j^t tiré selon la loi a posteriori de τ sachant Y_j et s_t.
  - $s_j^{t+1} = \delta_t s_j^t + (1 - \delta_t) A(Y_j, s_t)$.
- $S = \frac{1}{J} \sum_{j=1}^{J} s_j$

FIGURE: Erreur relative $\frac{1}{N^2}\left\|S_{SAEM} - s_0\right\|^2$ en fonction de $10 \times J$

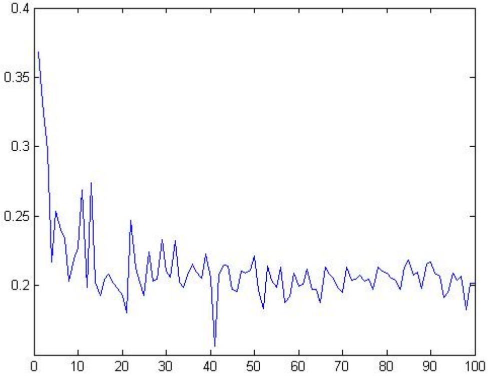
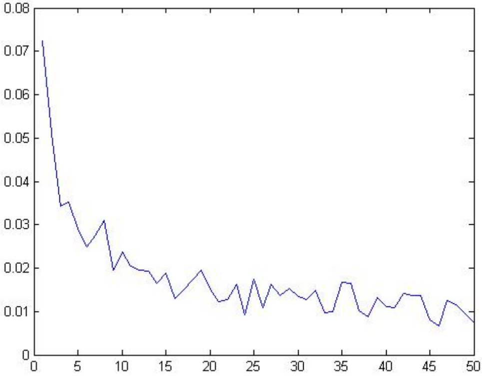


FIGURE: Erreur relative $\frac{1}{N^2}\left\|S_{SAEM} - s_0\right\|^2$ en fonction de $10 \times J$



Contribution au stage : théorème permettant de montrer la non convergence de l'algorithme MaxMax d'un point de vue statistique.

Théorème

Soit $\widehat{T}^n$ un estimateur quelconque des translations $(\tau_1, \cdots, \tau_n)$, nous avons :

$$\mathbb{E}[\frac{1}{n}\sum_{m=1}^{n}(\hat{T}_m^n - \tau_m)^2] \geq \sum_{k\neq 0} \frac{\epsilon^2}{|s'|_2^2 + (2\pi k)^2\epsilon^2}$$

- Cet algorithme a été construit pour débruiter des images à l'aide d'un moyennage non local de tous les pixels de l'image .
- Nous avons considéré un créneau bruité $y(i) = s_0(i) + \sigma(i)$

Principe de l'algorithme des non local means

Nous effectuons un moyennage pondéré de tous les points de l'image : $NL(y)(i) = \sum_{j \in I} w(i,j)y(j)$,

$w(i,j) = \frac{1}{Z(i)} e^{-\frac{\|y(N_i)-y(N_j)\|_{2,a}^2}{h^2}}$,

$Z(i) = \sum_{j \in I} e^{-\frac{\|y(N_i)-y(N_j)\|_{2,a}^2}{h^2}}$

- $y(N_i)$ est la restriction de y à une fenètre de similarité N_i.

- L'algorithme
  - Patch= voisinage d'un point.
  - Nous cherchons les patchs "proches" et nous les moyennons.
  - Nous recommençons pour tous les patchs du signal à débruiter.
- Quelques résultats

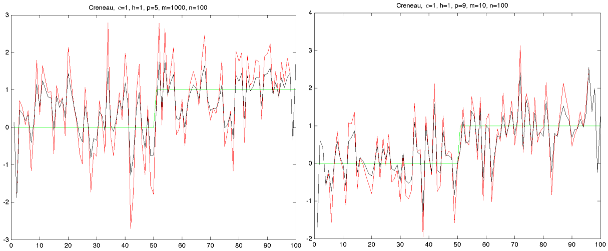


FIGURE: Application de l'algorithme des NL means pour deux valeurs de p et de J

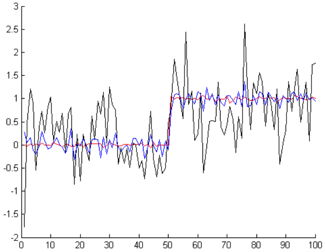


FIGURE: En rouge : EM ; en bleu : Max-Max ; en noir : Non Local Means.

- Étendre la preuve des points fixes aux voisinages de ceux-ci.
- Regarder de plus près l'erreur commise sur les translations lorsque celles-ci ne sont pas tirées de manière uniforme.

- Wu, C. F.Jeff, *On the Convergence Properties of the EM Algorithm*. Annals of Statistics, March 1983.
- S.Gallot, D.Hulin, J.Lafontaine, *Riemannian Geometry*. Second Edition, 1980.
- Christian P.Robert, *Le choix bayésien*. 2006.
- J.Bigot, S.Gadat, *A deconvolution approach to estimation of a common shape in a shifted curves model*. 2010.
- A.Buades, B.Coll, J.M.Morel, *Image Denoising Methods. A New Nonlocal Principle*. 2010
- J.Lafontaine, *Introduction aux variétés différentielles*. Nouvelle édition, 2010.

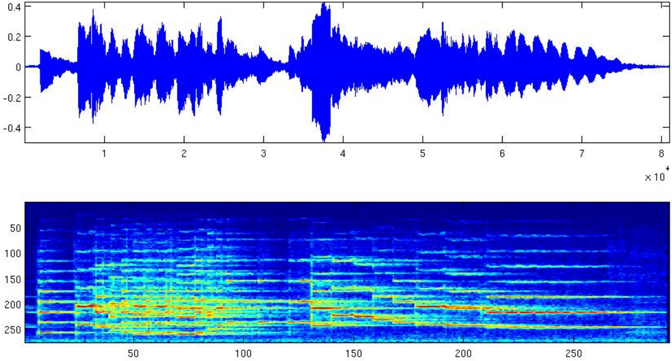
Débruitage des signaux audio par la méthode du seuillage par bloc

Marie d'Autume
Eric Martin
Christophe Varray

CMLA - Stage de L3
Sous la direction de
Jean-Michel Morel et Eva Theumann

Février - Juin 2012

Le passage dans le domaine temps-fréquence

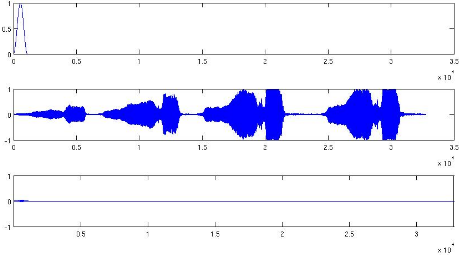


Représentation temps-fréquence du signal

Le passage dans le domaine temps-fréquence

STFT

Signal $\Longrightarrow$ spectrogramme.
fenêtrage $\Rightarrow$ DFT $\Rightarrow$ stockage.

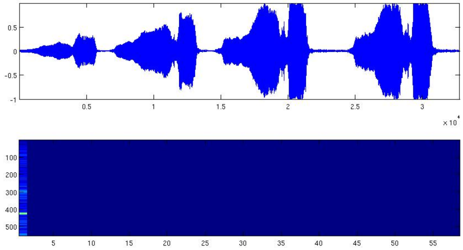


Remplissage du spectrogramme - une colonne remplie

Le passage dans le domaine temps-fréquence

STFT

Signal $\Longrightarrow$ spectrogramme.
fenêtrage $\Rightarrow$ DFT $\Rightarrow$ stockage.



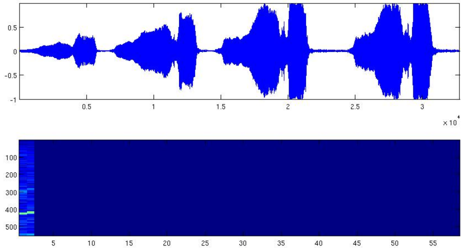
Remplissage du spectrogramme - une colonne remplie

Le passage dans le domaine temps-fréquence

STFT

Signal $\Longrightarrow$ spectrogramme.

fenêtrage $\Rightarrow$ DFT $\Rightarrow$ stockage.



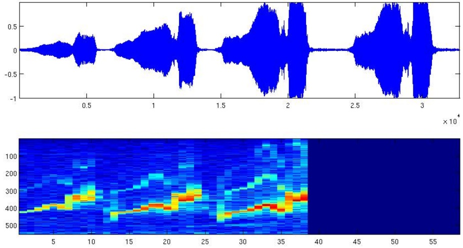
Remplissage du spectrogramme - deux colonnes remplies

Le passage dans le domaine temps-fréquence

STFT

Signal $\Longrightarrow$ spectrogramme.

fenêtrage $\Rightarrow$ DFT $\Rightarrow$ stockage.

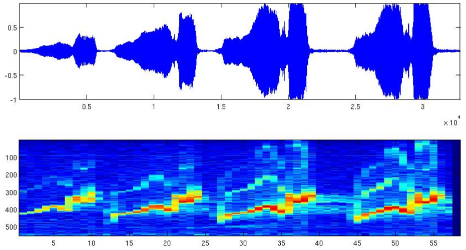


Remplissage du spectrogramme - rempli au $2/3$

Le passage dans le domaine temps-fréquence

STFT

Signal $\Longrightarrow$ spectrogramme.
fenêtrage $\Rightarrow$ DFT $\Rightarrow$ stockage.



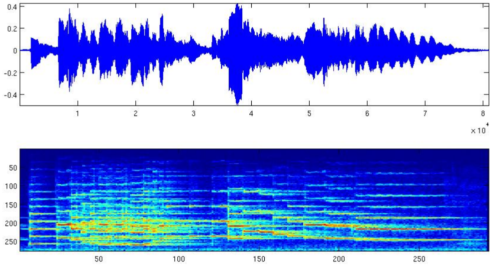
Remplissage du spectrogramme - Fin

Le passage dans le domaine temps-fréquence

STFT

Signal $\Longrightarrow$ spectrogramme.

fenêtrage $\Rightarrow$ DFT $\Rightarrow$ stockage.

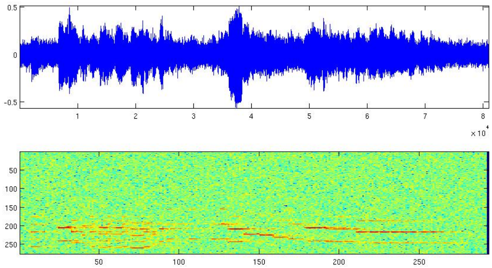


Représentation temps-fréquence du signal

Le passage dans le domaine temps-fréquence

STFT

Signal $\Longrightarrow$ spectrogramme.
fenêtrage $\Rightarrow$ DFT $\Rightarrow$ stockage.



Morceau de clarinette bruité par un bruit blanc

Le filtre de Wiener

- Notations
  - $y = f + \varepsilon \;\Rightarrow\; Y_{[l,k]} = F_{[l,k]} + \varepsilon_{[l,k]}$ dans le domaine temps-fréquence
    - ϵ bruit blanc gaussien : $\mathbb{E}(\varepsilon) = 0 \;\; \mathrm{Var}(\varepsilon) = \sigma^2$
  - $\hat{F}_{[l,k]} = a_{[l,k]} Y_{[l,k]}$: débruitage par convolution
- $\mathbb{E}[\sum_{k,l} \|\hat{F}_{[k,l]} - F_{[l,k]}\|^2]$: erreur quadratique moyenne

Le filtre de Wiener

- Notations
  - $y = f + \varepsilon \;\Rightarrow\; Y_{[l,k]} = F_{[l,k]} + \varepsilon_{[l,k]}$ dans le domaine temps-fréquence
    - ϵ bruit blanc gaussien : $\mathbb{E}(\varepsilon) = 0 \;\; \mathrm{Var}(\varepsilon) = \sigma^2$
  - $\hat{F}_{[l,k]} = a_{[l,k]} Y_{[l,k]}$: débruitage par convolution
- $\mathbb{E}[\sum_{k,l} \|\hat{F}_{[k,l]} - F_{[l,k]}\|^2]$: erreur quadratique moyenne

Le filtre de Wiener

- Notations
  - $y = f + \varepsilon \;\Rightarrow\; Y_{[l,k]} = F_{[l,k]} + \varepsilon_{[l,k]}$ dans le domaine temps-fréquence
    - ϵ bruit blanc gaussien : $\mathbb{E}(\varepsilon) = 0 \;\; \mathrm{Var}(\varepsilon) = \sigma^2$
  - $\hat{F}_{[l,k]} = a_{[l,k]} Y_{[l,k]}$: débruitage par convolution
- $\mathbb{E}[\sum_{k,l} \|\hat{F}_{[k,l]} - F_{[l,k]}\|^2]$: erreur quadratique moyenne

Le filtre de Wiener

- Notations
  - $y = f + \varepsilon \;\Rightarrow\; Y_{[l,k]} = F_{[l,k]} + \varepsilon_{[l,k]}$ dans le domaine temps-fréquence
    - ϵ bruit blanc gaussien : $\mathbb{E}(\varepsilon) = 0 \;\; \mathrm{Var}(\varepsilon) = \sigma^2$
  - $\hat{F}_{[l,k]} = a_{[l,k]} Y_{[l,k]}$: débruitage par convolution
- $\mathbb{E}[\sum_{k,l} \|\hat{F}_{[k,l]} - F_{[l,k]}\|^2]$: erreur quadratique moyenne

Filtre de Wiener idéal

Minimise l'erreur quadratique $a_{[l,k]} = \left(\frac{|F_{[l,k]}|^2}{\sigma^2+|F_{[l,k]}|^2}\right)$

suppose les $|F_{[l,k]}|$ connus

- Filtre de Wiener empirique
  - $a_{[l,k]} = \left(\frac{|Y_{[l,k]}|^2-\sigma^2}{|Y_{[l,k]}|^2}\right)_+$
- But : trouver une meilleure approximation des $|F_{[l,k]}|$

Le filtre de Wiener

- Notations
  - $y = f + \varepsilon \;\Rightarrow\; Y_{[l,k]} = F_{[l,k]} + \varepsilon_{[l,k]}$ dans le domaine temps-fréquence
    - ϵ bruit blanc gaussien : $\mathbb{E}(\varepsilon) = 0 \;\; \mathrm{Var}(\varepsilon) = \sigma^2$
  - $\hat{F}_{[l,k]} = a_{[l,k]} Y_{[l,k]}$: débruitage par convolution
- $\mathbb{E}[\sum_{k,l} \|\hat{F}_{[k,l]} - F_{[l,k]}\|^2]$: erreur quadratique moyenne

Filtre de Wiener idéal

Minimise l'erreur quadratique $a_{[l,k]} = \left(\frac{|F_{[l,k]}|^2}{\sigma^2 + |F_{[l,k]}|^2}\right)$

suppose les $|F_{[l,k]}|$ connus

- Filtre de Wiener empirique
  - $a_{[l,k]} = \left(\frac{|Y_{[l,k]}|^2 - \sigma^2}{|Y_{[l,k]}|^2}\right)_+$
- But : trouver une meilleure approximation des $|F_{[l,k]}|$

Le filtre de Wiener

- Notations
  - $y = f + \varepsilon \;\Rightarrow\; Y_{[l,k]} = F_{[l,k]} + \varepsilon_{[l,k]}$ dans le domaine temps-fréquence
    - ϵ bruit blanc gaussien : $\mathbb{E}(\varepsilon) = 0$ $\text{Var}(\varepsilon) = \sigma^2$
  - $\hat{F}_{[l,k]} = a_{[l,k]} Y_{[l,k]}$: débruitage par convolution
- $\mathbb{E}[\sum_{k,l} \| \hat{F}_{[k,l]} - F_{[l,k]} \|^2]$: erreur quadratique moyenne

Filtre de Wiener idéal

Minimise l'erreur quadratique $a_{[l,k]} = \left(\frac{|F_{[l,k]}|^2}{\sigma^2 + |F_{[l,k]}|^2} \right)$

suppose les $|F_{[l,k]}|$ connus

- Filtre de Wiener empirique
  - $a_{[l,k]} = \left(\frac{|Y_{[l,k]}|^2 - \sigma^2}{|Y_{[l,k]}|^2} \right)_+$
- But : trouver une meilleure approximation des $|F_{[l,k]}|$

Le seuillage par bloc

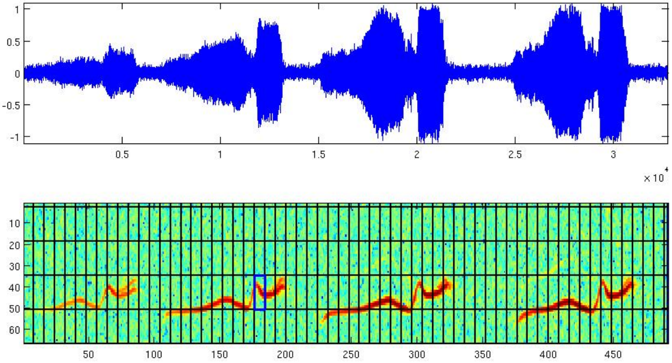


FIGURE: Découpage du spectrogramme en macro-blocs

Le seuillage par bloc

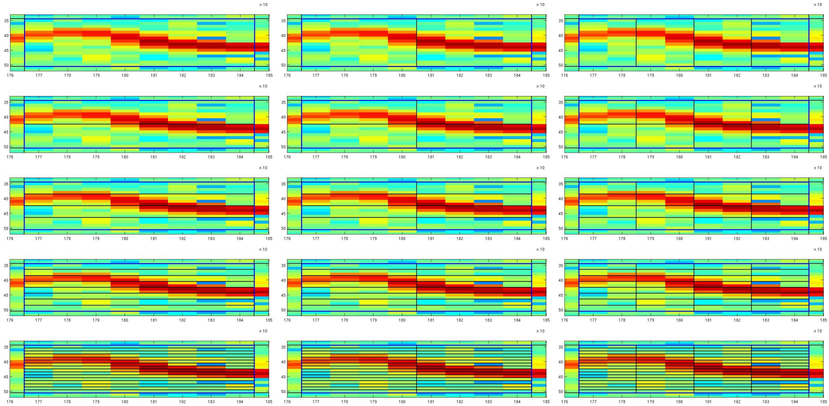


FIGURE: Découpages possibles

Le seuillage par bloc

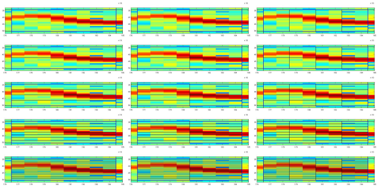


FIGURE: Découpage possibles

- $R_{Macro_{bloc}} = \sum_{1}^{Nb\ de\ mini_{blocs}} R_{mini_{bloc}i}$
- Théorème SURE $\Longrightarrow$ calcul du risque du bloc i
- Estimateur du signal obtenu
- Utiliser Wiener empirique avec notre estimateur

Le seuillage par bloc

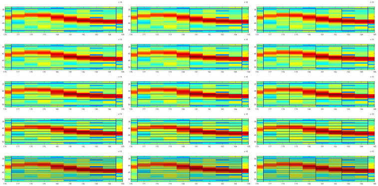


FIGURE: Découpage possibles

- $R_{Macro_{bloc}} = \sum_1^{Nb\ de\ mini_{blocs}} R_{mini_{bloc}i}$
- Théorème SURE $\Longrightarrow$ calcul du risque du bloc i
- Estimateur du signal obtenu
- Utiliser Wiener empirique avec notre estimateur

Le seuillage par bloc

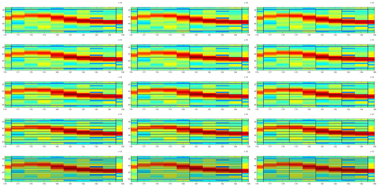


FIGURE: Découpage possibles

- $R_{Macro_{bloc}} = \sum_{1}^{Nb\ \ de\ \ mini_{blocs}} R_{mini_{bloc}i}$
- Théorème SURE $\Longrightarrow$ calcul du risque du bloc i
- Estimateur du signal obtenu
- Utiliser Wiener empirique avec notre estimateur

Le seuillage par bloc

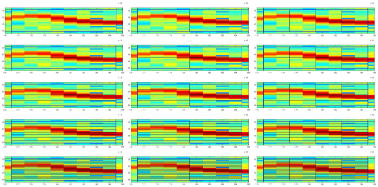


FIGURE: Découpage possibles

- $R_{Macro_{bloc}} = \sum_{1}^{Nb\ de\ mini_{blocs}} R_{mini_{bloc}i}$
- Théorème SURE $\Longrightarrow$ calcul du risque du bloc i
- Estimateur du signal obtenu
- Utiliser Wiener empirique avec notre estimateur

Critère d'évaluation des différentes méthodes et mis en place des benchs

la SNR : ratio signal sur bruit

Avec f_0 le signal original et f le signal débruité :

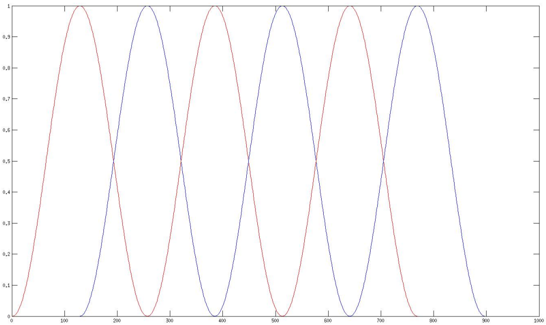
$$SNR = 10 * \log_{10}\left(\frac{\|f_0\|_{l^2}^2}{\|f_0 - f\|_{l^2}^2}\right)$$

Batterie de tests :

- Pour chaque $\sigma \longrightarrow$ bruitage du signal
  - pour chaque $size_{win} \longrightarrow$: test de toutes les méthodes et variantes.

Effet du facteur de redondance - Explications

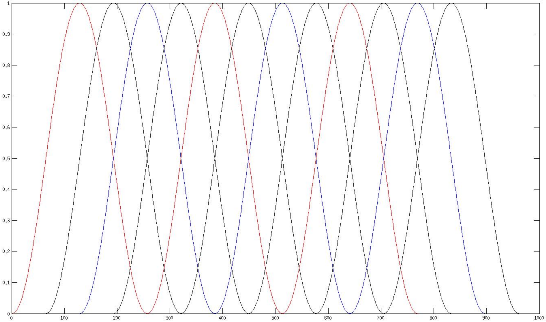
- Facteur logarithmique de redondance $k \implies$ recouvrement de 2^k



Superposition de $2^1 = 2$

Effet du facteur de redondance - Explications

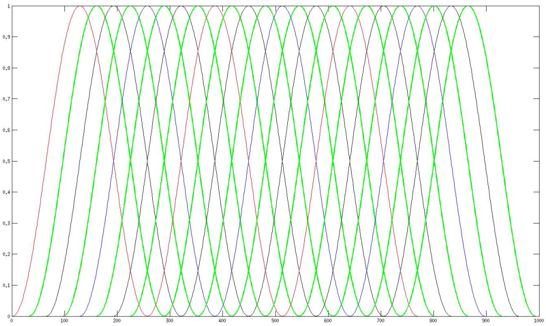
- Facteur logarithmique de redondance $k \implies$ recouvrement de 2^k



Superposition de $2^2 = 4$

Effet du facteur de redondance - Explications

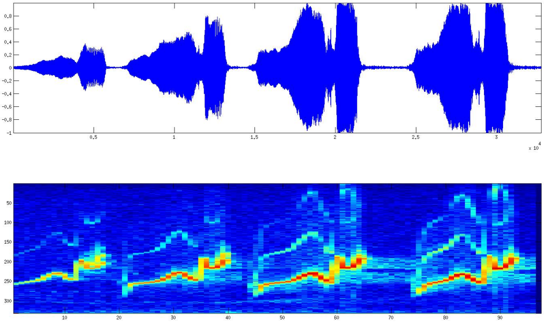
- Facteur logarithmique de redondance $k \implies$ recouvrement de 2^k



Superposition de $2^3 = 8$

Effet du facteur de redondance - Explications

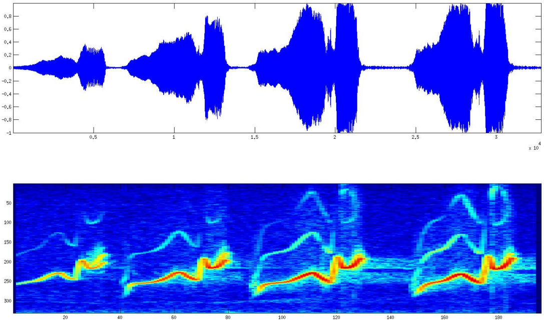
- Facteur logarithmique de redondance $k \implies$ recouvrement de 2^k



Superposition de $2^1 = 2$

Effet du facteur de redondance - Explications

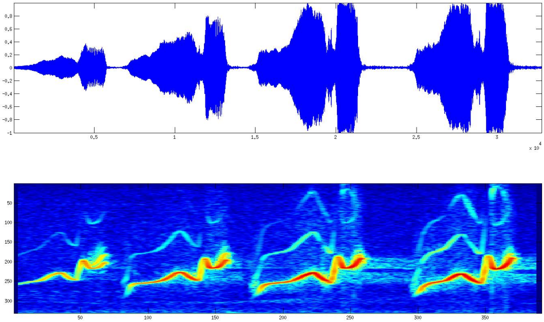
- Facteur logarithmique de redondance $k \implies$ recouvrement de 2^k



Superposition de $2^2 = 4$

Effet du facteur de redondance - Explications

- Facteur logarithmique de redondance $k \implies$ recouvrement de 2^k



Superposition de $2^3 = 8$

Effet du facteur de redondance - Résultats

Tableaux de SNR, pour différentes tailles de fenêtres utilisées :

Time_win = 115 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
K=1	29,393	25,248	22,837	21,21	19,766	18,688	17,725	17,025	16,32	15,621
K=2	29,794	25,5	23,19	21,505	20,093	19,001	18,048	17,361	16,721	15,949
K=3	29,975	25,704	23,361	21,664	20,294	19,19	18,255	17,537	16,884	16,143

- Améliorations quelque soit le niveau du bruit

Time_win = 175 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
K=1	28,371	24,83	22,633	20,959	19,688	18,622	17,741	17,02	16,395	15,624
K=2	28,277	24,882	22,769	21,225	19,864	18,837	18,045	17,281	16,617	15,987
K=3	27,833	24,72	22,714	21,233	19,9	18,911	18,095	17,399	16,766	16,093

- Taille de fenêtre plus grande $\Longrightarrow$ améliorations en grand bruit uniquement.

Effet du facteur de redondance - Résultats

Tableaux de SNR, pour différentes tailles de fenêtres utilisées :

Time_win = 115 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
K=1	29,393	25,248	22,837	21,21	19,766	18,688	17,725	17,025	16,32	15,621
K=2	29,794	25,5	23,19	21,505	20,093	19,001	18,048	17,361	16,721	15,949
K=3	29,975	25,704	23,361	21,664	20,294	19,19	18,255	17,537	16,884	16,143

- Améliorations quelque soit le niveau du bruit

Time_win = 175 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
K=1	28,371	24,83	22,633	20,959	19,688	18,622	17,741	17,02	16,395	15,624
K=2	28,277	24,882	22,769	21,225	19,864	18,837	18,045	17,281	16,617	15,987
K=3	27,833	24,72	22,714	21,233	19,9	18,911	18,095	17,399	16,766	16,093

- Taille de fenêtre plus grande $\Longrightarrow$ améliorations en grand bruit uniquement.

Effet du facteur de redondance - Résultats

Tableaux de SNR, pour différentes tailles de fenêtres utilisées :

Time_win = 115 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
K=1	29,393	25,248	22,837	21,21	19,766	18,688	17,725	17,025	16,32	15,621
K=2	29,794	25,5	23,19	21,505	20,093	19,001	18,048	17,361	16,721	15,949
K=3	29,975	25,704	23,361	21,664	20,294	19,19	18,255	17,537	16,884	16,143

- Améliorations quelque soit le niveau du bruit

Time_win = 175 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
K=1	28,371	24,83	22,633	20,959	19,688	18,622	17,741	17,02	16,395	15,624
K=2	28,277	24,882	22,769	21,225	19,864	18,837	18,045	17,281	16,617	15,987
K=3	27,833	24,72	22,714	21,233	19,9	18,911	18,095	17,399	16,766	16,093

- Taille de fenêtre plus grande $\Longrightarrow$ améliorations en grand bruit uniquement.

Effet du facteur de redondance - Résultats

Tableaux de SNR, pour différentes tailles de fenêtres utilisées :

Time_win = 115 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
K=1	29,393	25,248	22,837	21,21	19,766	18,688	17,725	17,025	16,32	15,621
K=2	29,794	25,5	23,19	21,505	20,093	19,001	18,048	17,361	16,721	15,949
K=3	29,975	25,704	23,361	21,664	20,294	19,19	18,255	17,537	16,884	16,143

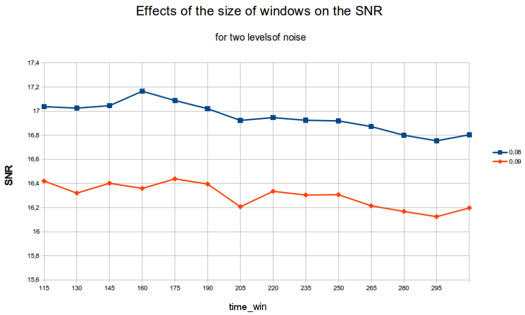
- Améliorations quelque soit le niveau du bruit

Time_win = 175 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
K=1	28,371	24,83	22,633	20,959	19,688	18,622	17,741	17,02	16,395	15,624
K=2	28,277	24,882	22,769	21,225	19,864	18,837	18,045	17,281	16,617	15,987
K=3	27,833	24,72	22,714	21,233	19,9	18,911	18,095	17,399	16,766	16,093

- Taille de fenêtre plus grande $\implies$ améliorations en grand bruit uniquement.

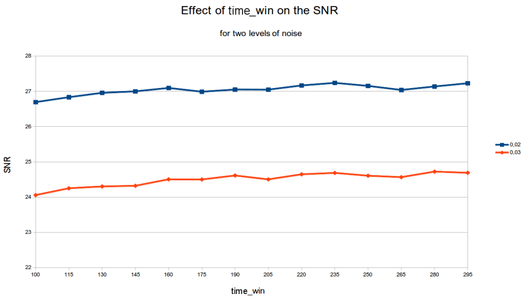
Effet de la taille de fenêtre - Résultats



$$f_{sampling} = 11kHz$$

- Taille optimale :
  - dépend de $f_{sampling}$
  - ne dépend pas de σ

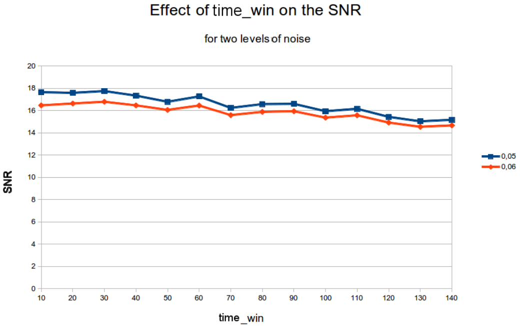
Effet de la taille de fenêtre - Résultats



$$f_{sampling} = 16kHz$$

- Taille optimale :
  - dépend de $f_{sampling}$
  - ne dépend pas de σ

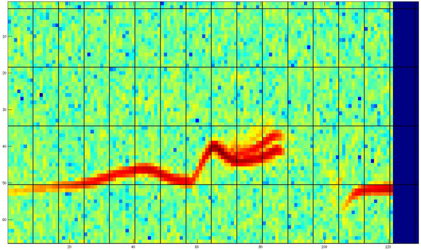
Effet de la taille de fenêtre - Résultats



$$f_{sampling} = 41 kHz$$

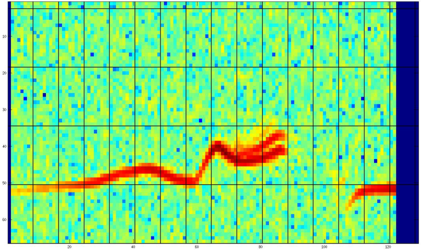
- Taille optimale :
  - dépend de $f_{sampling}$
  - ne dépend pas de σ

Méthode invariante en temps - Explication de la méthode



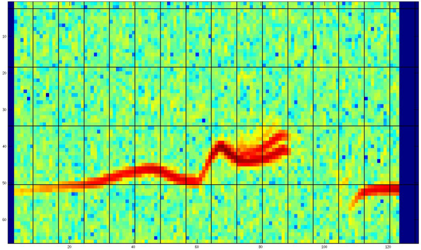
- Calculer la STFT du signal : $STFT_{coef} \leftarrow STFT(f)$
- Pour tout j de 0 à 7 faire :
  - Ajouter j colonnes nulles au début de la matrice $STFT_{coef}$
  - Débruiter avec l'algorithme non modifié du seuillage par block.
  - Stocker le signal reconstitué
- Renvoyer la moyenne des 8 signaux ainsi calculés

Méthode invariante en temps - Explication de la méthode



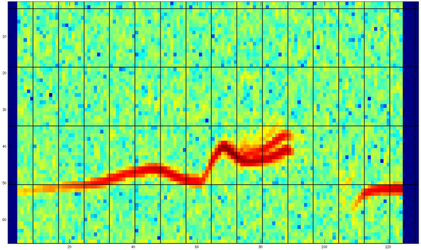
- Calculer la STFT du signal : $STFT_{coef} \leftarrow STFT(f)$
- Pour tout j de 0 à 7 faire :
  - Ajouter j colonnes nulles au début de la matrice $STFT_{coef}$
  - Débruiter avec l'algorithme non modifié du seuillage par block.
  - Stocker le signal reconstitué
- Renvoyer la moyenne des 8 signaux ainsi calculés

Méthode invariante en temps - Explication de la méthode



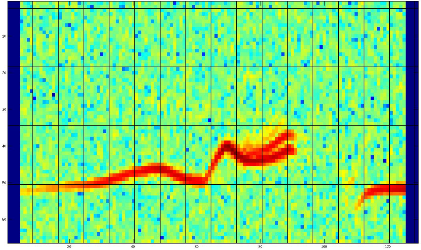
- Calculer la STFT du signal : $STFT_{coef} \leftarrow STFT(f)$
- Pour tout j de 0 à 7 faire :
  - Ajouter j colonnes nulles au début de la matrice $STFT_{coef}$
  - Débruiter avec l'algorithme non modifié du seuillage par block.
  - Stocker le signal reconstitué
- Renvoyer la moyenne des 8 signaux ainsi calculés

Méthode invariante en temps - Explication de la méthode



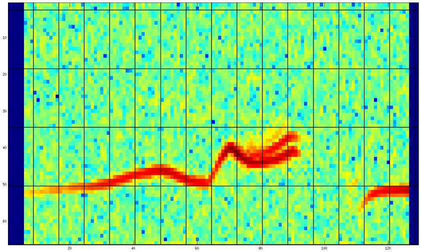
- Calculer la STFT du signal : $STFT_{coef} \leftarrow STFT(f)$
- Pour tout j de 0 à 7 faire :
  - Ajouter j colonnes nulles au début de la matrice $STFT_{coef}$
  - Débruiter avec l'algorithme non modifié du seuillage par block.
  - Stocker le signal reconstitué
- Renvoyer la moyenne des 8 signaux ainsi calculés

Méthode invariante en temps - Explication de la méthode



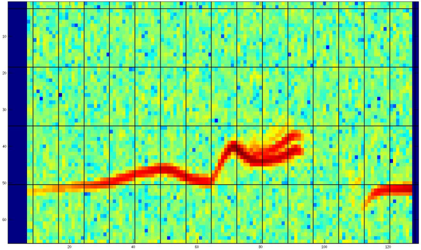
- Calculer la STFT du signal : $STFT_{coef} \leftarrow STFT(f)$
- Pour tout j de 0 à 7 faire :
  - Ajouter j colonnes nulles au début de la matrice $STFT_{coef}$
  - Débruiter avec l'algorithme non modifié du seuillage par block.
  - Stocker le signal reconstitué
- Renvoyer la moyenne des 8 signaux ainsi calculés

Méthode invariante en temps - Explication de la méthode



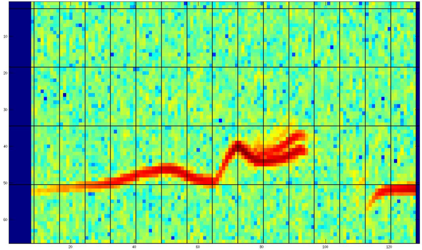
- Calculer la STFT du signal : $STFT_{coef} \leftarrow STFT(f)$
- Pour tout j de 0 à 7 faire :
  - Ajouter j colonnes nulles au début de la matrice $STFT_{coef}$
  - Débruiter avec l'algorithme non modifié du seuillage par block.
  - Stocker le signal reconstitué
- Renvoyer la moyenne des 8 signaux ainsi calculés

Méthode invariante en temps - Explication de la méthode



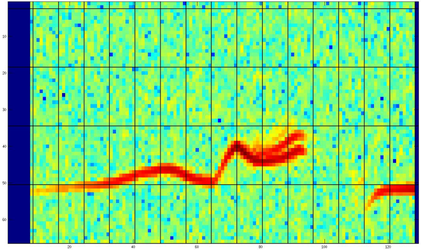
- Calculer la STFT du signal : $STFT_{coef} \leftarrow STFT(f)$
- Pour tout j de 0 à 7 faire :
  - Ajouter j colonnes nulles au début de la matrice $STFT_{coef}$
  - Débruiter avec l'algorithme non modifié du seuillage par block.
  - Stocker le signal reconstitué
- Renvoyer la moyenne des 8 signaux ainsi calculés

Méthode invariante en temps - Explication de la méthode



- Calculer la STFT du signal : $STFT_{coef} \leftarrow STFT(f)$
- Pour tout j de 0 à 7 faire :
  - Ajouter j colonnes nulles au début de la matrice $STFT_{coef}$
  - Débruiter avec l'algorithme non modifié du seuillage par block.
  - Stocker le signal reconstitué
- Renvoyer la moyenne des 8 signaux ainsi calculés

Méthode invariante en temps - Explication de la méthode



- Calculer la STFT du signal : $STFT_{coef} \leftarrow STFT(f)$
- Pour tout j de 0 à 7 faire :
  - Ajouter j colonnes nulles au début de la matrice $STFT_{coef}$
  - Débruiter avec l'algorithme non modifié du seuillage par block.
  - Stocker le signal reconstitué
- Renvoyer la moyenne des 8 signaux ainsi calculés

Méthode invariante en temps - Résultats

Time_win = 115 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	29,393	25,248	22,837	21,21	19,766	18,688	17,725	17,025	16,32	15,621
Inv	29,441	25,32	22,989	21,337	19,89	18,832	17,854	17,159	16,453	15,767
Gain	0,048	0,072	0,152	0,127	0,124	0,144	0,129	0,134	0,133	0,146

Time_win = 50 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	20,714	19,688	18,627	17,615	16,783	16,054	15,378	14,809	14,354	13,87
Inv	20,73	19,741	18,722	17,747	16,923	16,18	15,517	14,969	14,475	14,028
Gain	0,016	0,053	0,095	0,132	0,14	0,126	0,139	0,16	0,121	0,158

- Gain SNR minime
- En revanche : → Écoute :
  - Morceau de piano, basse et percussion
  - Morceau de trompettes et batteries

Méthode invariante en temps - Résultats

Time_win = 115 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	29,393	25,248	22,837	21,21	19,766	18,688	17,725	17,025	16,32	15,621
Inv	29,441	25,32	22,989	21,337	19,89	18,832	17,854	17,159	16,453	15,767
Gain	0,048	0,072	0,152	0,127	0,124	0,144	0,129	0,134	0,133	0,146

Time_win = 50 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	20,714	19,688	18,627	17,615	16,783	16,054	15,378	14,809	14,354	13,87
Inv	20,73	19,741	18,722	17,747	16,923	16,18	15,517	14,969	14,475	14,028
Gain	0,016	0,053	0,095	0,132	0,14	0,126	0,139	0,16	0,121	0,158

- Gain SNR minime
- En revanche : $\rightarrow$ Écoute :
  - Morceau de piano, basse et percussion
  - Morceau de trompettes et batteries

Méthode invariante en temps - Résultats

Time_win = 115 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	29,393	25,248	22,837	21,21	19,766	18,688	17,725	17,025	16,32	15,621
Inv	29,441	25,32	22,989	21,337	19,89	18,832	17,854	17,159	16,453	15,767
Gain	0,048	0,072	0,152	0,127	0,124	0,144	0,129	0,134	0,133	0,146

Time_win = 50 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	20,714	19,688	18,627	17,615	16,783	16,054	15,378	14,809	14,354	13,87
Inv	20,73	19,741	18,722	17,747	16,923	16,18	15,517	14,969	14,475	14,028
Gain	0,016	0,053	0,095	0,132	0,14	0,126	0,139	0,16	0,121	0,158

- Gain SNR minime
- En revanche : $\rightarrow$ Écoute :
  - Morceau de piano, basse et percussion
  - Morceau de trompettes et batteries

Méthode invariante en temps - Résultats

Time_win = 115 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	29,393	25,248	22,837	21,21	19,766	18,688	17,725	17,025	16,32	15,621
Inv	29,441	25,32	22,989	21,337	19,89	18,832	17,854	17,159	16,453	15,767
Gain	0,048	0,072	0,152	0,127	0,124	0,144	0,129	0,134	0,133	0,146

Time_win = 50 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	20,714	19,688	18,627	17,615	16,783	16,054	15,378	14,809	14,354	13,87
Inv	20,73	19,741	18,722	17,747	16,923	16,18	15,517	14,969	14,475	14,028
Gain	0,016	0,053	0,095	0,132	0,14	0,126	0,139	0,16	0,121	0,158

- Gain SNR minime
- En revanche : $\rightarrow$ Écoute :
  - Morceau de piano, basse et percussion
  - Morceau de trompettes et batteries

Méthode invariante en temps - Résultats

Time_win = 115 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	29,393	25,248	22,837	21,21	19,766	18,688	17,725	17,025	16,32	15,621
Inv	29,441	25,32	22,989	21,337	19,89	18,832	17,854	17,159	16,453	15,767
Gain	0,048	0,072	0,152	0,127	0,124	0,144	0,129	0,134	0,133	0,146

Time_win = 50 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	20,714	19,688	18,627	17,615	16,783	16,054	15,378	14,809	14,354	13,87
Inv	20,73	19,741	18,722	17,747	16,923	16,18	15,517	14,969	14,475	14,028
Gain	0,016	0,053	0,095	0,132	0,14	0,126	0,139	0,16	0,121	0,158

- Gain SNR minime
- En revanche : $\rightarrow$ Écoute :
  - Morceau de piano, basse et percussion
  - Morceau de trompettes et batteries

Méthode invariante en temps - Interprétation

Time_win = 115 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	29,393	25,248	22,837	21,21	19,766	18,688	17,725	17,025	16,32	15,621
Inv	29,441	25,32	22,989	21,337	19,89	18,832	17,854	17,159	16,453	15,767
Gain	0,048	0,072	0,152	0,127	0,124	0,144	0,129	0,134	0,133	0,146

Time_win = 50 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	20,714	19,688	18,627	17,615	16,783	16,054	15,378	14,809	14,354	13,87
Inv	20,73	19,741	18,722	17,747	16,923	16,18	15,517	14,969	14,475	14,028
Gain	0,016	0,053	0,095	0,132	0,14	0,126	0,139	0,16	0,121	0,158

- Bruit musicaux $\Longrightarrow$ très localisés
  - Pire des cas : Aucune amélioration
  - Meilleur des cas : chaque bruit divisé par 8
- Donc : bruit plus diffus
  - Mais encore présent $\Longrightarrow$ peu d'évolution SNR.
  - Cependant mieux réparti $\Longrightarrow$ écoute plus agréable.

Méthode invariante en temps - Interprétation

Time_win = 115 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	29,393	25,248	22,837	21,21	19,766	18,688	17,725	17,025	16,32	15,621
Inv	29,441	25,32	22,989	21,337	19,89	18,832	17,854	17,159	16,453	15,767
Gain	0,048	0,072	0,152	0,127	0,124	0,144	0,129	0,134	0,133	0,146

Time_win = 50 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	20,714	19,688	18,627	17,615	16,783	16,054	15,378	14,809	14,354	13,87
Inv	20,73	19,741	18,722	17,747	16,923	16,18	15,517	14,969	14,475	14,028
Gain	0,016	0,053	0,095	0,132	0,14	0,126	0,139	0,16	0,121	0,158

- Bruit musicaux $\Longrightarrow$ très localisés
  - Pire des cas : Aucune amélioration
  - Meilleur des cas : chaque bruit divisé par 8
- Donc : bruit plus diffus
  - Mais encore présent $\Longrightarrow$ peu d'évolution SNR.
  - Cependant mieux réparti $\Longrightarrow$ écoute plus agréable.

Méthode invariante en temps - Interprétation

Time_win = 115 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	29,393	25,248	22,837	21,21	19,766	18,688	17,725	17,025	16,32	15,621
Inv	29,441	25,32	22,989	21,337	19,89	18,832	17,854	17,159	16,453	15,767
Gain	0,048	0,072	0,152	0,127	0,124	0,144	0,129	0,134	0,133	0,146

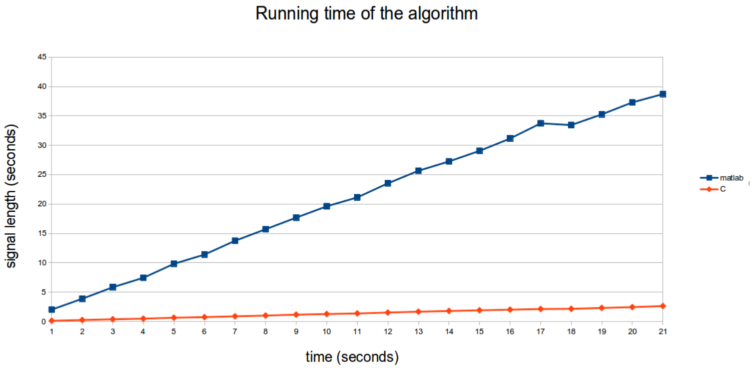
Time_win = 50 ms

sigma	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
Classic	20,714	19,688	18,627	17,615	16,783	16,054	15,378	14,809	14,354	13,87
Inv	20,73	19,741	18,722	17,747	16,923	16,18	15,517	14,969	14,475	14,028
Gain	0,016	0,053	0,095	0,132	0,14	0,126	0,139	0,16	0,121	0,158

- Bruit musicaux $\Longrightarrow$ très localisés
  - Pire des cas : Aucune amélioration
  - Meilleur des cas : chaque bruit divisé par 8
- Donc : bruit plus diffus
  - Mais encore présent $\Longrightarrow$ peu d'évolution SNR.
  - Cependant mieux réparti $\Longrightarrow$ écoute plus agréable.

Mise du code en C

Temps de calcul $= f$ (taille du signal)



- Bleu : matlab
- Rouge : code C

Mise en commun des améliorations

Pour chaque niveau de bruit et taille de fenêtre, 6 méthodes à disposition : $k = 1, 2$ *ou* 3 pour une méthode invariante en temps ou non.

Tableau contenant, pour un signal donné, la meilleure méthode pour *time_win* et σ donnés :

	time_win	Comparaison : Max snr										
σ	0	0	0,01	0,02	0,03	0,04	0,05	0,06	0,07	0,08	0,09	0,1
	0	0	0	0	0	0	0	0	0	0	0	0
	100	42,231	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3
	115	42,704	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3
	130	39,077	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3
	145	38,468	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3
	160	36,923	inv_k2	inv_k2	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3
	175	33,258	inv_k1	inv_k2	inv_k2	inv_k2	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3
	190	32,639	inv_k1	inv_k1	inv_k2	inv_k2	inv_k2	inv_k3	inv_k3	inv_k3	inv_k3	inv_k3
	205	32,13	inv_k1	inv_k1	inv_k1	inv_k2	inv_k2	inv_k2	inv_k2	inv_k3	inv_k3	inv_k3
	220	30,02	inv_k1	inv_k1	inv_k1	inv_k1	inv_k2	inv_k2	inv_k2	inv_k2	inv_k2	inv_k3
	235	28,75	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k2	inv_k2	inv_k2	inv_k2
	250	27,171	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k2	inv_k2	inv_k2
	265	26,204	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k2
	280	24,892	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1
	295	24,458	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1	inv_k1

Mise en commun des améliorations

Pour chaque niveau de bruit et taille de fenêtre, 6 méthodes à disposition : $k = 1, 2$ *ou* 3 pour une méthode invariante en temps ou non.
Tableau contenant les SNR utilisant les différentes méthodes pour deux couples $(time\_win, \sigma)$ donnés :

	Sigma=0,05 et time_win=130 ms	Sigma=0,08 et time_win=160 ms
K=1	19,744	17,088
inv et k=1	19,895	17,155
K=2	20,067	17,381
inv et k=2	20,164	17,424
K=3	20,246	17,48
inv et k=3	20,296	17,55

Conclusions et perspectives

- Théorème mis en œuvre très puissant
    - Applications encore nombreuses
    - Exemple : Changer la manière de découper (Fine to coarse) pour plus d'améliorations
- Ce dont on dispose :
    - Plusieurs méthodes et paramètres
    - Choix de paramètres optimaux dépend du type de son
    - En pratique, le SNR pas connu donc choix subjectif entre les différents résultats.

Conclusions et perspectives

- Théorème mis en œuvre très puissant
  - Applications encore nombreuses
  - Exemple : Changer la manière de découper (Fine to coarse) pour plus d'améliorations
- Ce dont on dispose :
  - Plusieurs méthodes et paramètres
  - Choix de paramètres optimaux dépend du type de son
  - En pratique, le SNR pas connu donc choix subjectif entre les différents résultats.

Conclusions et perspectives

- Théorème mis en œuvre très puissant
  - Applications encore nombreuses
  - Exemple : Changer la manière de découper (Fine to coarse) pour plus d'améliorations
- Ce dont on dispose :
  - Plusieurs méthodes et paramètres
  - Choix de paramètres optimaux dépend du type de son
  - En pratique, le SNR pas connu donc choix subjectif entre les différents résultats.

Conclusions et perspectives

- Théorème mis en œuvre très puissant
  - Applications encore nombreuses
  - Exemple : Changer la manière de découper (Fine to coarse) pour plus d'améliorations
- Ce dont on dispose :
  - Plusieurs méthodes et paramètres
  - Choix de paramètres optimaux dépend du type de son
  - En pratique, le SNR pas connu donc choix subjectif entre les différents résultats.

Conclusions et perspectives

- Théorème mis en œuvre très puissant
  - Applications encore nombreuses
  - Exemple : Changer la manière de découper (Fine to coarse) pour plus d'améliorations
- Ce dont on dispose :
  - Plusieurs méthodes et paramètres
  - Choix de paramètres optimaux dépend du type de son
  - En pratique, le SNR pas connu donc choix subjectif entre les différents résultats.

Remerciement : Eva Theumann et Jean-Michel Morel

Conclusions et perspectives

- Théorème mis en œuvre très puissant
  - Applications encore nombreuses
  - Exemple : Changer la manière de découper (Fine to coarse) pour plus d'améliorations
- Ce dont on dispose :
  - Plusieurs méthodes et paramètres
  - Choix de paramètres optimaux dépend du type de son
  - En pratique, le SNR pas connu donc choix subjectif entre les différents résultats.

Remerciement : Eva Theumann et Jean-Michel Morel

À bientôt sur Ipol ! $\rightarrow$ IPOL IMAGE PROCESSING ON LINE

Fenêtres de Hanning, formules et justifications

Fenêtre de Hanning

On définit la fenêtre de Hanning par :

$$\begin{array}{rccc} h: & \mathbb{R} & \to & \mathbb{R} \\ & x & \mapsto & \begin{cases} 0 & \text{si } |x| \geq 1 \\ \frac{1+\cos(\pi x)}{2} & \text{si } x \in [-1,1] \end{cases} \end{array}$$

On a alors la l-ième fenêtre qui est :

$$w_l(j) = w\left(j - lq\right) = \frac{1 + \cos\left(\frac{\pi t}{q} - l\pi\right)}{2}$$

sur l'intervalle de points $\{(l-1)\,q, (l+1)\,q\}$

Fenêtres de Hanning, formules et justifications

On a alors la l-ième fenêtre qui est :

$$w_l(j) = w\left(j - lq\right) = \frac{1 + \cos\left(\frac{\pi t}{q} - l\pi\right)}{2}$$

sur l'intervalle de points $\left\{\left(l-1\right)q, \left(l+1\right)q\right\}$

Propriété pour la reconstruction

Pour tout $n \in \{q, ..., N-q\}$, on a

$$\sum_{l=1}^{K} w_l(n) = 1$$

Stein Unbiased Risk Estimator (SURE)

- Y vecteur aléatoire gaussien de dimension p.
  - $\mathrm{cov(Y)} = \mathrm{Id}$ et $\mathbb{E}[Y] = F$
  - $Y + h(Y) = aY = \hat{F}$ un estimateur de F

Théorème

Si $\mathbb{E}[\sum_{j=1}^{p} |\frac{\partial h_j}{\partial Y_j}(Y)|] < \infty$ alors $\hat{R} = p + \|h(Y)\|_2^2 + 2div(h(Y))$ est un estimateur sans biais du risque R de $\hat{F}$.

$$\hat{R}_i = \bar{\sigma}_i^2 \left(B_i^{\#} + \frac{\lambda^2 B_i^{\#} - 2\lambda(B_i^{\#} - 2)}{\frac{\overline{Y_i^2}}{\bar{\sigma}_i^2}} \mathbb{1}_{\overline{Y_i^2} \geqslant \lambda \bar{\sigma}_i^2} + B_i^{\#} \left(\frac{\overline{Y_i^2}}{\bar{\sigma}_i^2} - 2 \right) \mathbb{1}_{\overline{Y_i^2} < \lambda \bar{\sigma}_i^2} \right)$$

.

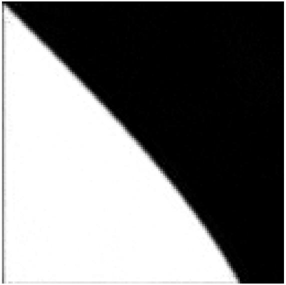
Débruitage d'images : vers l'optimum ?

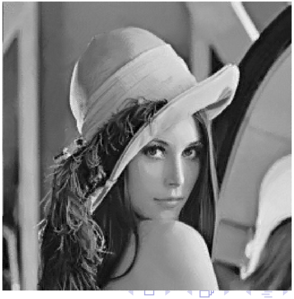
S.ABELARD, S.AMAR, E.STRICKLER

Encadrants : J-M.MOREL, M.DELBRACIO

Mardi 3 Juillet

Premier exemple de débruitage :





Parcours de recherche :

A mi-parcours :

- ✓ Etude de divers algorithmes
- ✓ Calculs du risque minimax de ces algorithmes
- ✓ Implémentation de l'algorithme PCA-Wiener

Travail complété depuis par :

- ✓ Vérification et rédaction du Graal (optimum en $n^{-4/3}$)
- ✓ Vérification et rédaction de l'optimalité de l'Anisotropic NLM
- ✓ Etude (et modification) du code de l'ANLM.
- ✓ Rédaction d'errata concernant ANLM (envoyés aux auteurs)

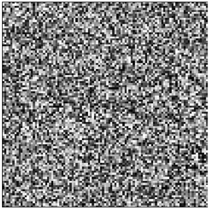
Quelques notations :

- u, P : image idéale

- $\tilde{u}$, $\tilde{P}$: image bruitée

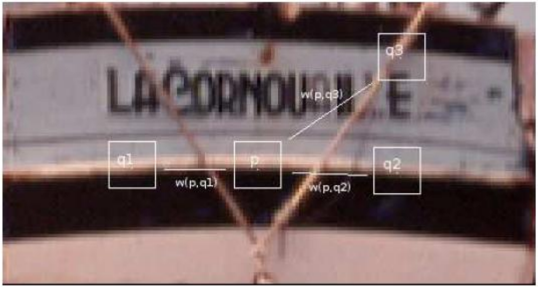
- $\hat{u}$, $\hat{P}$: image débruitée

- $\tilde{u} - u$: bruit gaussien



“Patch-based algorithm” :

- On recherche le plus d’échantillons possibles pour moyenner :
- il y a des patchs similaires dans une image
- ... mais aussi dans le monde !
- on tire partie de la “redondance” ou “auto-similarité” des images naturelles



A.Levin, B.Nadler, Natural Image Denoising : Optimality and Inherent Bounds, 2011

Optimalité quadratique de la méthode Bayesienne :

Critère d'erreur :

On définit l'erreur quadratique de moyenne (**MMSE**)

$$MMSE(\tilde{P}) = \mathbb{E}(\| P - \hat{P} \|^2 | \tilde{P})$$

Le théorème qui suit est ***fondamental***, il nous donne un premier résultat d'optimalité, sous réserve de la connaissance de la densité de probabilité $\mathbb{P}(P)$.

Théorème :

Pour un patch bruité $\tilde{P}$, l'erreur quadratique de moyenne (**MMSE**) est minimisée par :

$$\hat{P} = \mathbb{E}(P \mid \tilde{P}) = \int \mathbb{P}(P \mid \tilde{P}) P dP = \int \frac{\mathbb{P}(\tilde{P} \mid P)\mathbb{P}(P)}{\mathbb{P}(\tilde{P})} P dP$$

Algorithme Shotgun NL-means :

On rappelle que :

$$\hat{P} = \int \frac{\mathbb{P}(\tilde{P} \mid P)\mathbb{P}(P)}{\mathbb{P}(\tilde{P})} P dP$$

- Entrée : $N = 10^{10}$ patchs P_i sans bruit.
- Pour tout $\tilde{P}$ extrait de l'image bruitée,

$$\hat{P} = \frac{\sum \mathbb{P}(\tilde{P} \mid P_i) P_i}{\sum \mathbb{P}(\tilde{P} \mid P_i)} \quad \text{où} \quad \mathbb{P}(\tilde{P} \mid P_i) = (2\pi\sigma^2)^{-h^2/2} e^{-\|P_i - \tilde{P}\|^2/2\sigma^2}.$$

- Idée : récolter le plus grand nombre d'échantillons de patchs (10 milliards ! !) pour calculer une intégrale (méthode de Monte-Carlo). Augmenter la taille du patch fait croître le nombre d'échantillons à considérer pour l'image.

A.Levin, B.Nadler, Natural Image Denoising : Optimality and Inherent Bounds, 2011

Image débruitée par la méthode shotgun :

(a) Original image

(b) Noisy input

(c) Opt. MMSE, PSNR=23.93dB

(d) BM3D, PSNR=23.86dB

Premières définitions :

Risque :

On définit ici le risque quadratique :

$$R(\hat{u}, u) = \mathbb{E}\left(||\hat{u} - u||_2^2\right)$$

Biais :

$$b^2 = ||\mathbb{E}(\hat{u}) - u||_2^2$$

Variance :

$$\sigma^2 = \mathbb{E}\left(||\hat{u} - \mathbb{E}(\hat{u})||_2^2\right)$$

Propriété :

$$R(\hat{u}, u) = b^2 + \sigma^2$$

Risque minimax :

- Comment faire disparaître la dépendance en u ?
- On choisit de prendre le pire cas afin de pouvoir garantir certaines bornes.

Risque minimax et estimateur du minimax :

On définit comme suit le *risque minimax* de l'algorithme $\mathcal{A}$ sur la classe $\mathcal{C}$:

$$R(\mathcal{A}) = \sup_{u \in \mathcal{C}} \mathbb{E}\left(||\hat{u}_{\mathcal{A}} - u||_2^2\right)$$

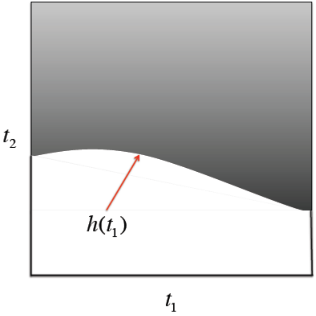
On cherche donc

$$\mathcal{A} = \arg\min_{\mathcal{A} \in A} R(\mathcal{A})$$

Avec A l'ensemble de ***tous*** les algorithmes possibles et $\hat{u}_{\mathcal{A}}$ l'estimateur hérité de l'algorithme $\mathcal{A}$. On cherche donc l'algorithme dont le pire cas est le moins catastrophique.

- Que choisir pour $\mathcal{C}$?

Un exemple de classe horizon :



En moyennant, on aura des soucis aux bords, en fonction de la régularité de ceux-ci.

Expression de u :

L'image considérée est du type :

$$u_h(t_1, t_2) = \mathbf{1}_{0 \leq t_2 < h(t_1)}(t_1, t_2)$$

Avec h une fonction à valeurs réelles que l'on prendra dans $\mathcal{C}^2(L)$, l'ensemble des fonctions à dérivée bornée par 1 et à dérivée seconde bornée par L (classe horizon 2)

Le *Saint Graal* du débruitage :

- Le théorème qui suit est ***fondamental*** car il nous donne une limite infranchissable (dont on se rapproche)

Théorème :

Soit une image de taille n. Sur la classe horizon définie précédemment, le risque minimax se comporte en

$$n^{-\frac{4}{3}}$$

- Attention : faire croître n revient à mieux échantillonner l'image, et non pas à l'agrandir !

A.P.Korostelev, A.B.Tsybakov, Minimax theory of image reconstruction, Springer-Verlag, 1993.

Sous-optimalité du NLM :

- Comparaison des diverses méthodes :

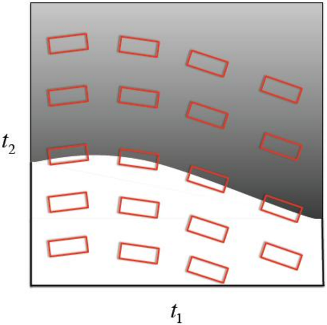
Estimateur localement polynomial	$\Theta(n^{-\frac{2}{3}})$
Semi-Yaroslavsky Filter	$\Theta(n^{-\frac{2}{3}})$
Non-Local Means	$O\left(\frac{\log^{\frac{1}{2}+\varepsilon} n}{n}\right)$
Optimalité	$\Theta(n^{-\frac{4}{3}})$

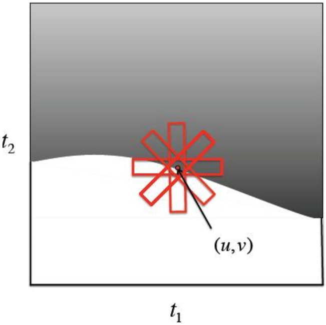
- Toutes sont sous-optimales
- On va modifier le NLM pour le rendre optimal.

R.G.Baraniuk, M.Narayan, A.Maleki, Suboptimality of standard Non Local Means denoising, preprint, 2011.

Optimisations du NLM :

- On remplace les patchs carrés par des rectangles suivant le contour
- Comment trouver la bonne orientation des rectangles ?
- GANLM : on utilise le gradient
- DANLM : on optimise sur un nombre fini d'angles
- **Théorème :** Ces deux algorithmes sont en $\Theta\left(n^{-\frac{4}{3}} \log^{\frac{4}{3}} n\right)$

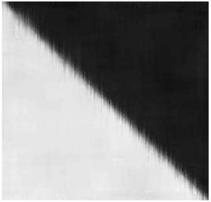




Importance de l'orientation :

(a) 0°, PSNR = 20.6dB

(b) 45°, PSNR = 19.2dB



(a) 90°, PSNR = 20.6dB

(d) 135°, PSNR = 28.9dB

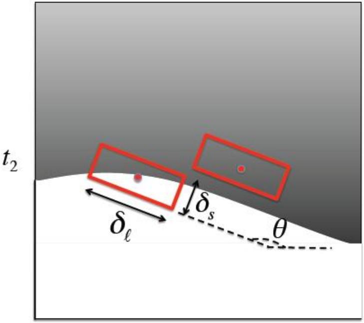
- Le danger réside aux bords !
- D'où l'intérêt de la classe horizon...
- et l'idée de prendre des *shape adaptive patchs*
- Dans ce cas, ce sont des rectangles
- Le choix de la taille reste un point à débattre.

Algorithme GANLM

Entrée :

$\hat{u}$: estimation de l'image
δ_l, δ_s : taille des voisinages
λ : seuil

Estimer le gradient de l'image (g_h, g_v)
Pour chaque pixel (i, j) faire :
$g(i,j) = \sqrt{g_h(i,j)^2 + g_v(i,j)^2}$
$\theta_{i,j} = \arctan\left(\frac{g_v(i,j)}{g_h(i,j)}\right)$



Si $g(i,j) \geq \lambda$:

appliquer ANLM avec les paramètres $\theta_{i,j}$, δ_l, δ_s

Sinon :

appliquer NLM avec $\delta = \sqrt{\delta_l \delta_s}$

Conclusion : le débruitage, un problème bientôt résolu ?

- Nous avons démontré l'optimalité du ANLM.
- Visuellement, sa supériorité n'est pas évidente.
- Il faut améliorer l'algorithme...
- en particulier l'ajustement de ses paramètres
- Mais peut-être aussi revenir sur nos critères...
- Voire considérer des apports d'autres domaines