Exemples d'étude de la stabilité numérique dans des logiciels de calcul scientifique

Jocelyne Erhel
INRIA-RENNES – PROJET ALADIN

Premier cas

code industriel d'éléments finis

1^{er} cas : extrait du code

```
DET(A, B, BC, D, E, F, G, H, AI) =
G * B * F + A * E * AI + D * H * BC - AI * B * D - BC * E * G - F * H * A

UN = 1
ZER = 0

DELTA = DET(RR1, Z1, UN, RR2, Z2, UN, R3, Z3, UN)
A2 = DET(UN, Z1, UN, ZER, Z2, UN, ZER, Z3, UN) / DELTA
B2 = DET(RR1, UN, UN, RR2, ZER, UN, R3, ZER, UN) / DELTA
D2 = DET(RR1, Z1, UN, RR2, Z2, ZER, R3, Z3, ZER) / DELTA

DT1 = + 2 * (A1 * B2 - A2 * B1)
DT2 = + (D1 * B2 - D2 * B1)
XCT1 = A1 * A2
XCT2 = B1 * B2
XCT3 = A1 * B2 + A2 * B1
XCT4 = A1 * D2 + A2 * D1
XCT5 = B1 * D2 + B2 * D1
XCT6 = D1 * D2
```

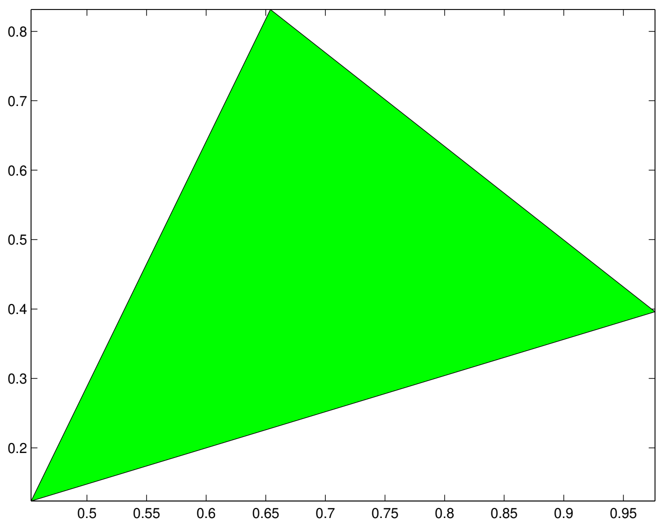
1^{er} cas : analyse mathématique du code

DET : calcul d'un déterminant 3×3

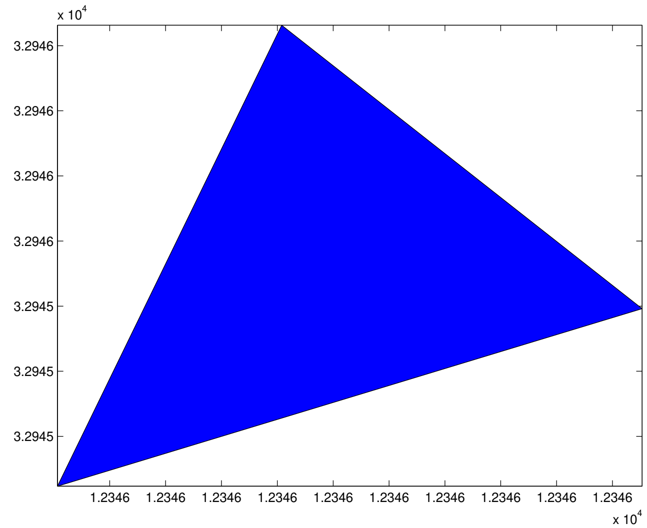
DELTA : calcul de l'aire du triangle de sommets $(rr1,z1)$,$(rr2,z2)$,$(rr3,z3)$

expérience : translation d'un triangle d'environ $(10^4,10^4)$

calcul en double précision



aire calculée = 0.15787634025000

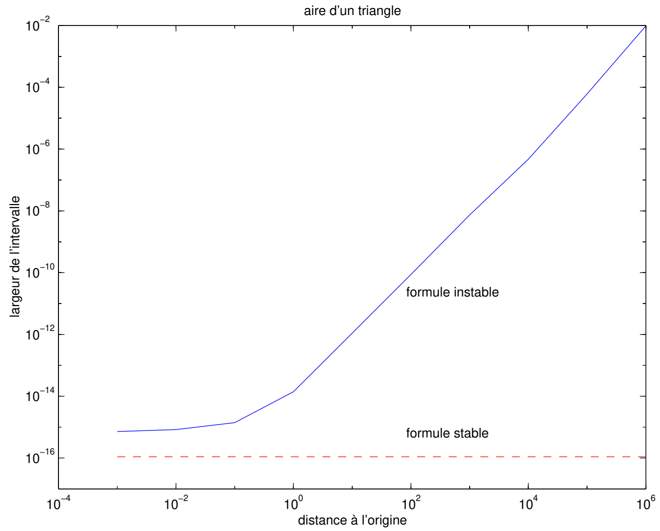


aire calculée = 0.15787628293037

1^{er} cas : analyse numérique du code

$$A = \pm \frac{1}{2} \times \begin{vmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{vmatrix}$$

calcul en
arithmétique d'intervalles



Formule instable : cancellations

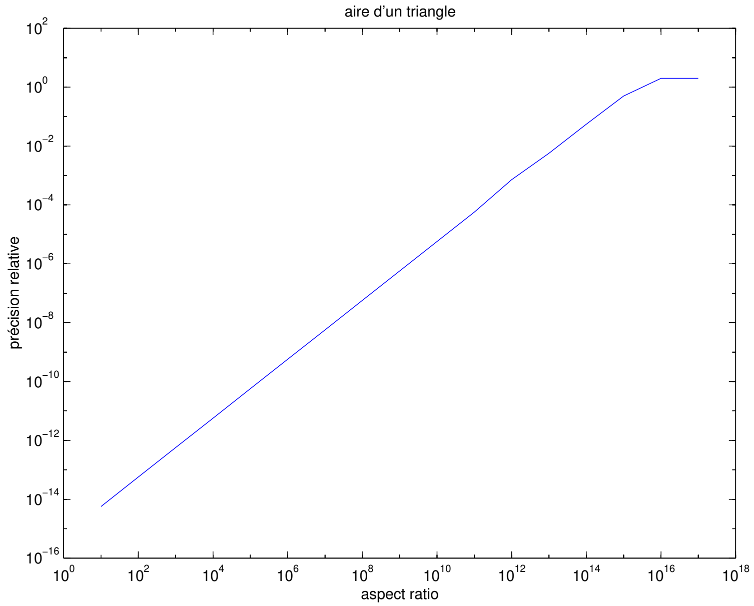
$$A = 1/2 \times |x_1 y_2 + x_2 y_3 + x_3 y_1 - x_1 y_3 - x_2 y_1 - x_3 y_2|$$

Formule stable : erreur en $O(\epsilon)$

$$A = \frac{1}{2} \times |(x_2 - x_1)(y_3 - y_1) - (y_2 - y_1)(x_3 - x_1)|$$

1^{er} cas : analyse du conditionnement

L'aire d'un triangle de base a et de hauteur h est $A = \frac{1}{2} \times a \times h$
On a $|\Delta A|/A \leq (1 + a/h) \max(|\Delta a|/a, |\Delta h|/a)$
où a/h est l'aspect ratio du triangle



calcul en arithmétique d'intervalles par la formule stable

Le calcul de l'aire d'un triangle aplati est mal conditionné

Les logiciels de maillage contrôlent l'aspect ratio des triangles

Deuxième cas

code industriel de convection

étude réalisée avec Stéphanie Rault

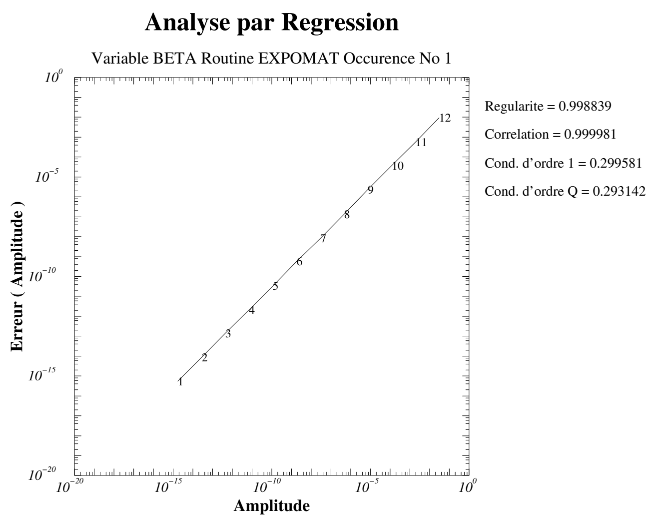
2^e cas : extrait du code

```
do 6 i=1,nn
gamma(i,i) = 1.d0
do 7 j=i-1,1,-1
gamma(i,j) = gamma(i,j+1)*alam(j)/(alam(j)-alam(i))
7 continue
6 continue

do 10 irn=1,nn

do 15 jrn=1,irn

tconv = zl * omg * r / ud
beta(irn,jrn) = dexp(-alam(irn)*tconv)*gamma(irn,jrn)
do 50 k=jrn,irn-1
beta(irn,jrn) = beta(irn,jrn)-beta(k,jrn)*gamma(irn,k)
50 continue
call somme(beta(irn,jrn),c0,tc0,ntc0,1.d0,cl,tcl,ncl,cl,tcl,ncl)
15 continue

10 continue
```

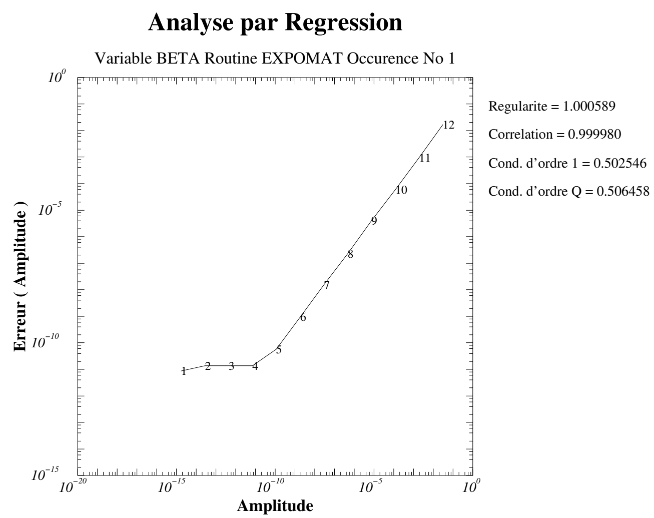
2^e cas : expériences numériques

Analyse statistique avec perturbations aléatoires des données

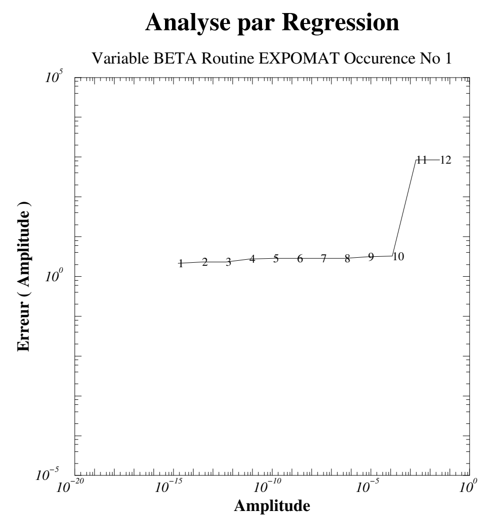
exemple 1



exemple 2



exemple 3



Le calcul est instable dans les exemples 2 et 3

2^e cas : analyse mathématique et numérique

Le code calcule
$$\left\{\begin{array}{l}
\varphi(t) = e^{At_c}\varphi_e(t-t_c) \\
t_c = \frac{\omega RL}{U} \\
A = \begin{pmatrix}
-\lambda_1 & 0 & \cdots & \cdots & \cdots & 0 \\
\lambda_1 & -\lambda_2 & 0 & \cdots & \cdots & 0 \\
0 & \lambda_2 & -\lambda_3 & \ddots & & \vdots \\
\vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\
\vdots & & \ddots & \ddots & \ddots & 0 \\
0 & \cdots & \cdots & 0 & \lambda_{n-1} & -\lambda_n
\end{pmatrix}
\end{array}\right.$$

Le code calcule l'exponentielle de matrice avec une diagonalisation

$$A = VDV^{-1} \quad et \quad \exp^{tA} = V \exp^{tD} V^{-1}$$

Calcul instable : au premier ordre

$$\frac{\|\Delta(\exp^{tA})\|}{\|\exp^{tA}\|} \leq 2cond(V)\frac{\|\Delta(V)\|}{\|V\|}$$

2^e cas : choix d'une méthode stable

Méthode de Padé avec "scaling and squaring" :
approximation mais calcul stable

exemple	1	2	3
$\| \exp_D^{tA} - \exp_P^{tA} \| / \| \exp_P^{tA} \|$	10^{-14}	10^{-10}	1
$cond(V)$	5	10^6	10^{19}

Troisième cas

code universitaire de simulation

d'écoulements

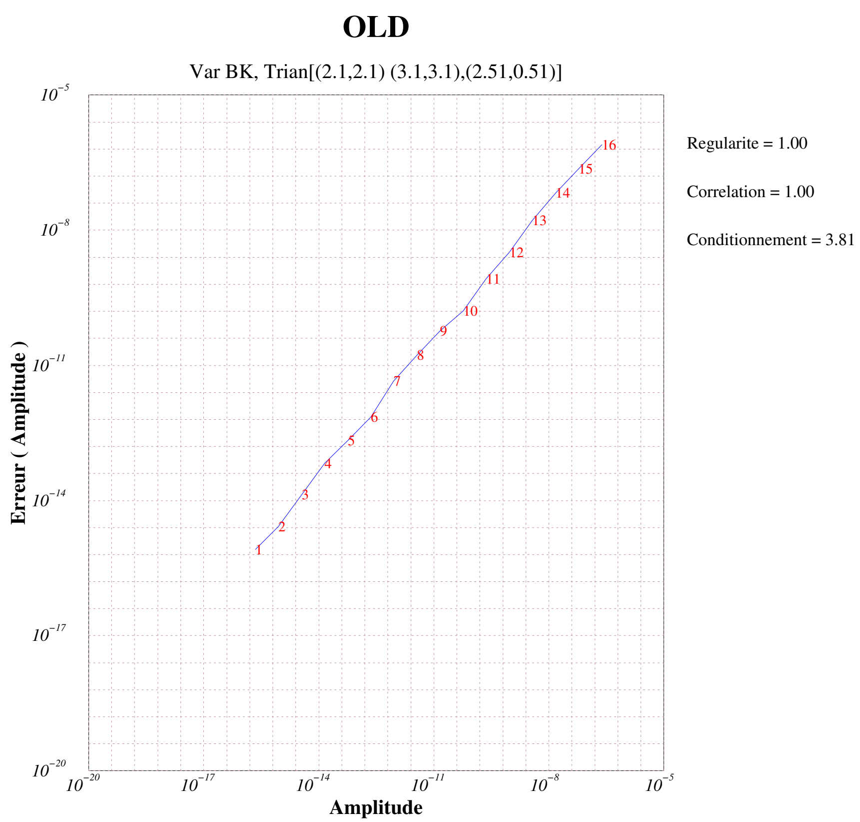
étude réalisée avec Hussein Hoteit

3^e cas : extrait du code

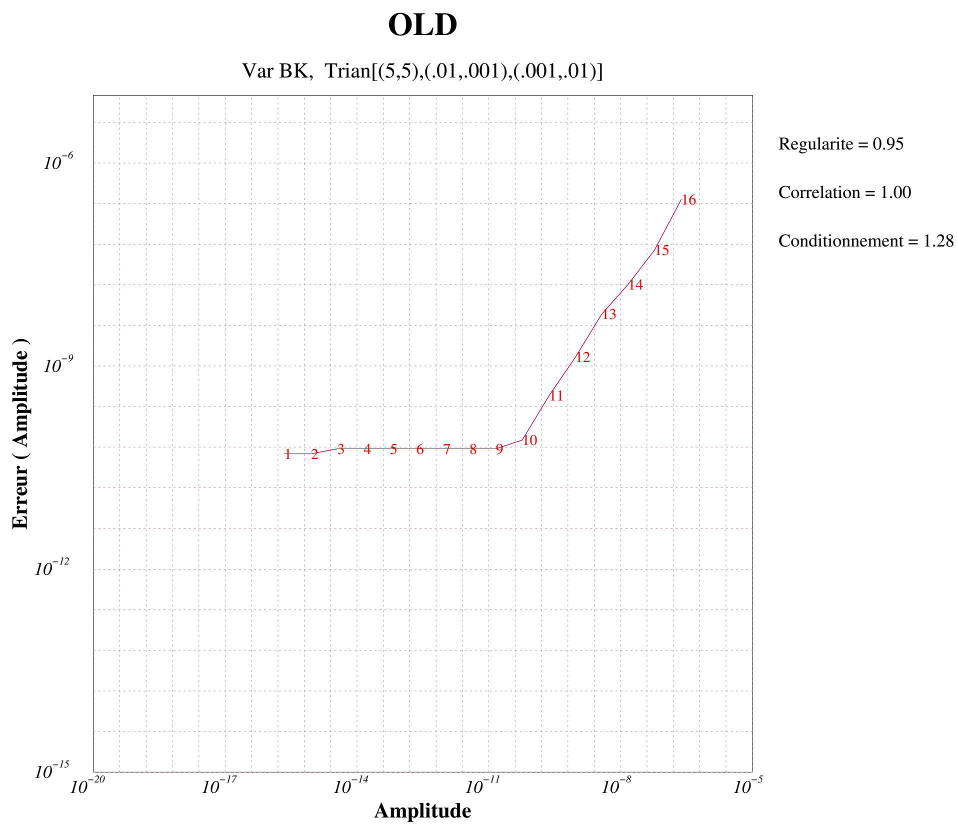
```
A=p2(1)-p1(1) ; B=p3(1)-p1(1) ; C=p2(2)-p1(2) ; D=p3(2)-p1(2)
detj=A*D-C*B
a2=A*A ; a3=a2*A ; a4=a3*A ; b2=B*B ; b3=b2*B ; b4=b3*B
c2=C*C ; c3=c2*C ; c4=c3*C ; d2=D*D ; d3=D*d2 ; d4=D*d3
kx2=KX*KX ; ky2=KY*KY ; kxy=KY*KX
a3b=a3*B ; b3a=b3*A ; c3d=c3*D ; d3c=d3*C ; ab2=a2*b2 ; cd2=c2*d2
ab=A*B ; cd=C*D ; abcd=ab*cd
deno=((a4*d2-a3b*(2*cd+d2)+ab2*(c2+2*cd+d2)-b3a*(c2+2*cd)
+b4*c2)*KY+(a2*(cd2-d3c+d4)+2*ab*(cd2-c3d-d3c)+b2*(c4-c3d+cd2))
KX)/detj
if (deno.eq.0) deno=1
bk(1,1)=(2*((a4-a3b+ab2)*ky2+(a2*(2*c2-cd+3*d2)+c2*(3*b2-ab)
-4*abcd)*kxy+(c4-c3d+cd2)*kx2))/deno
bk(1,2)=-(2*((a3b-ab2+b3a)*ky2+(a2*(cd-2*d2)+c2*(ab-2*b2)+
2*abcd+ab*d2+b2*cd)*kxy+(c3d-cd2+d3c)*kx2))/deno
bk(1,3)=-(2*((a4-2*a3b+2*ab2-b3a)*ky2+(a2*(2*c2-2*cd-d2)
-c2*(b2+2*ab)-ab*d2+6*abcd-b2*cd)*kxy+(c4-2*c3d+2*cd2-d3c)*kx2))/deno
bk(2,2)=(2*((ab2-b3a+b4)*ky2+(d2*(3*a2-ab)-4*abcd+b2*(3*c2-cd
+2*d2))*kxy+(cd2-d3c+d4)*kx2))/deno
bk(2,3)=(2*((a3b-2*ab2-b4+2*b3a)*ky2+(a2*(cd+d2)+ab*(c2+2*d2)
-6*abcd+b2*(c2+2*cd-2*d2))*kxy+(c3d-2*cd2+2*d3c-d4)*kx2))/deno
bk(3,3)=(2*((a4-3*a3b+4*ab2-3*b3a+b4)*ky2+(a2*(2*c2-3*cd+4*d2
)-ab*(3*c2+3*d2)+b2*(4*c2-3*cd+2*d2))*kxy+(4*cd2-3*c3d+c4+d4-3*
d3c)*kx2))/deno
```

3^e cas : expériences numériques (1)

exemple 1



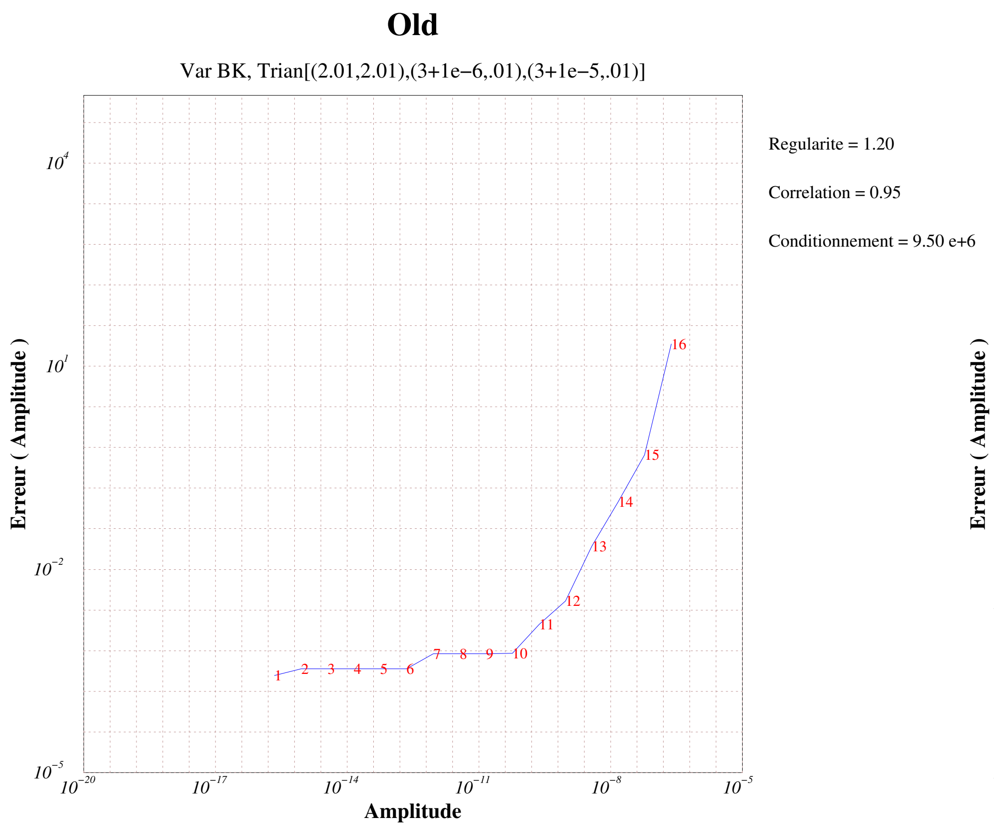
exemple 2



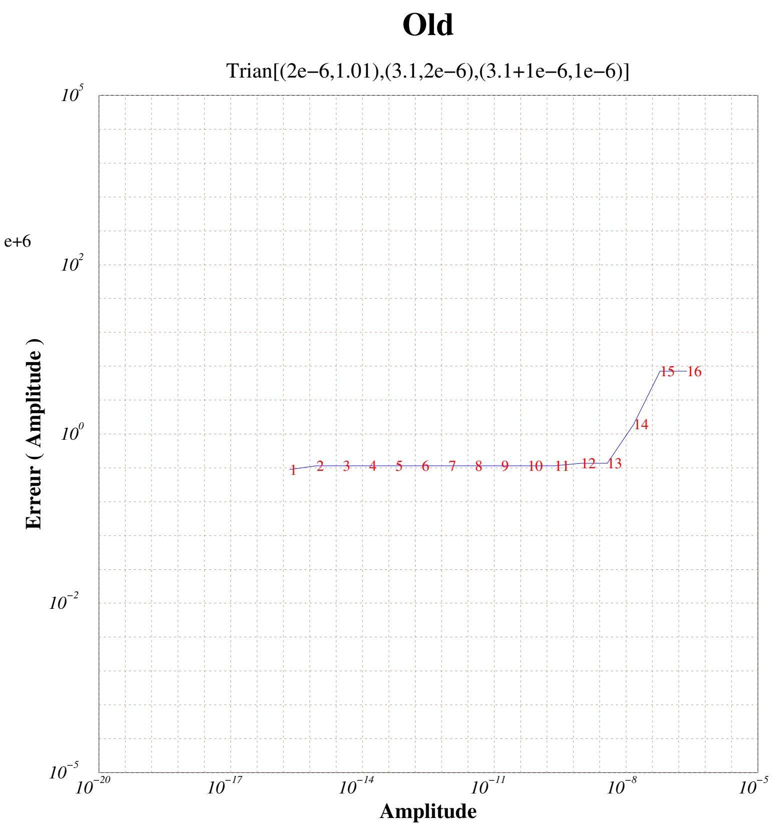
Dans les 2 exemples, le problème est bien conditionné
Dans l'exemple 2, le calcul est instable

3^e cas : expériences numériques (2)

exemple 3



exemple 4



Dans les 2 exemples, le problème est mal conditionné
et le calcul est instable

3^e cas : analyse mathématique

Le code calcule B^{-1} où $B = (B_{ij}) \in \mathbb{R}^{3\times 3}$

$$B_{ij} = \int_T w_i^T K^{-1} w_j dxdy$$
$$J = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$$
$$B_{ij} = \frac{1}{det(J)} \int_{\widehat{T}} \widehat{w}_i^T J^T K^{-1} J \widehat{w}_j dxdy$$

Le code d'origine génère les formules de B^{-1} par calcul formel
mais l'algorithme est instable

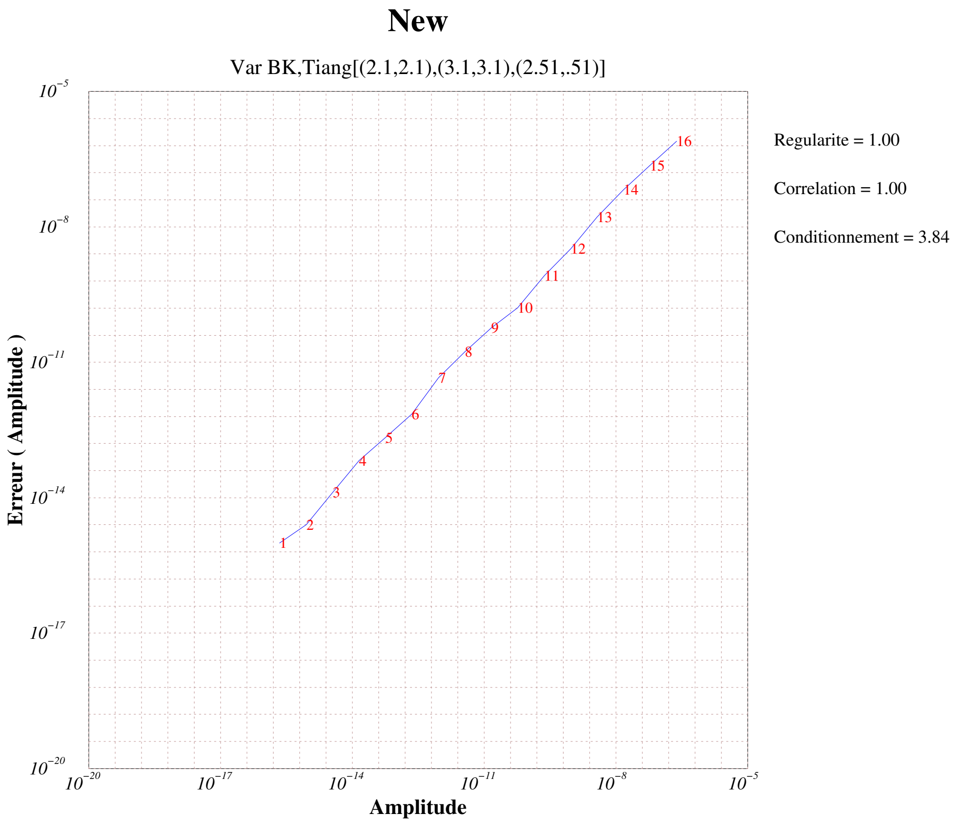
Toujours avec calcul formel, on génère les formules de la factorisation
$B^{-1} = L^{-T} D^{-1} L^{-1}$
Il faut prouver la stabilité de l'algorithme

3^e cas : nouveau code

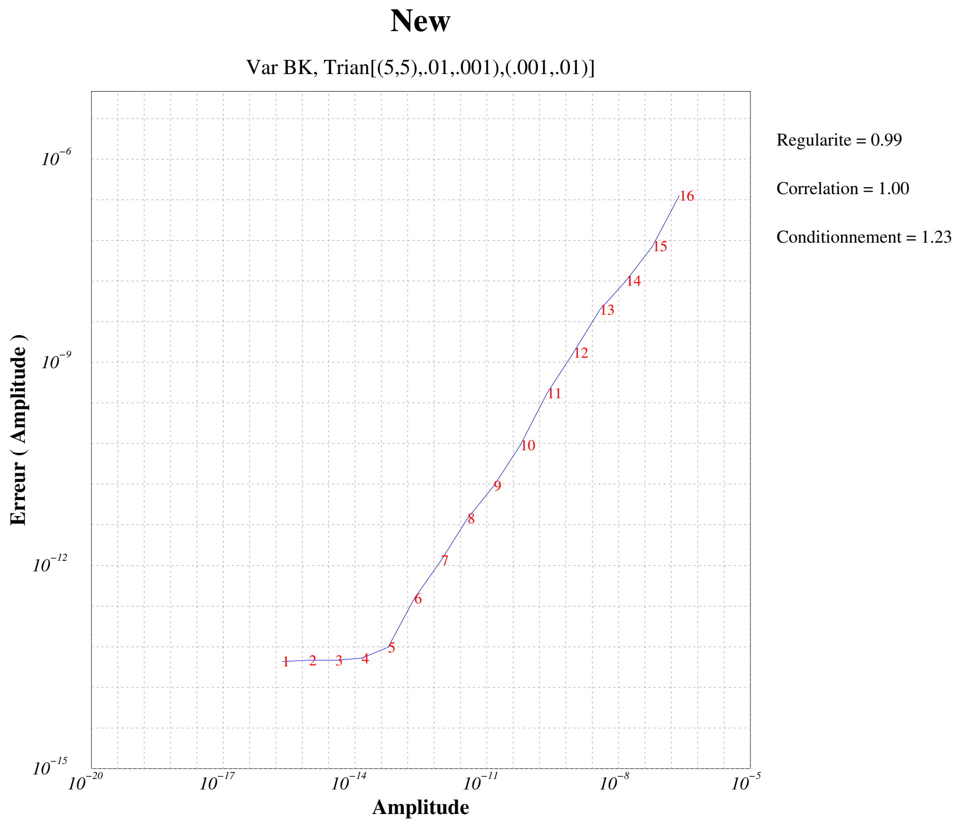
```
A=p2(1)-p1(1) ; B=p3(1)-p1(1) ; C=p2(2)-p1(2) ; D=p3(2)-p1(2)
detj=A*D-C*B
a2=a*a ; b2=b*b ; c2=c*c ; d2=d*d
ab=a*b ; cd=c*d ; abcd=ab*cd ; kxy=kx*ky
v1=(a2 - ab + b2)*ky*ky ; v2=(c2 - cd + d2)*kx*kx
deno=(v1/ky+v2/kx)*detj/2.
bk(1,1)=(c2*v2+(3*b2*c2-ab*c2-4*abcd+a2*(2*c2-cd+3*d2))*kxy+a2*v1)/deno
bk(1,2)=-(cd*v2+(a2*cd-2*a2*d2-2*b2*c2+b2*cd+ab*(c2+2*cd+d2))*kxy+ab*v1)/deno
bk(1,3)=-((c2-cd)*v2+(2*a2*c2-2*ab*c2-b2*c2-2*a2*cd+6*abcd-b2*cd-a2*d2-ab*d2)
*kxy+(a2-ab)*v1)/deno
bk(2,2)=(d2*v2+(3*a2*d2-4*abcd-ab*d2+b2*(3*c2-cd+2*d2))*kxy+b2*v1)/deno
bk(2,3)=((cd-d2)*v2+(a2*cd+a2*d2+b2*(c2+2*cd-2*d2)+ab*(c2-6*cd+2*d2))
*kxy+(ab-b2)*v1)/deno
bk(3,3)=((c2-2*cd+d2)*v2+(b2*(4*c2-3*cd+2*d2)-3*ab*(c2+d2)+a2*(2*c2-3*cd+4*d2))
*kxy+(a2-2*ab+b2)*v1)/deno
```

3^e cas : résultats avec le nouveau code - (1)

exemple 1



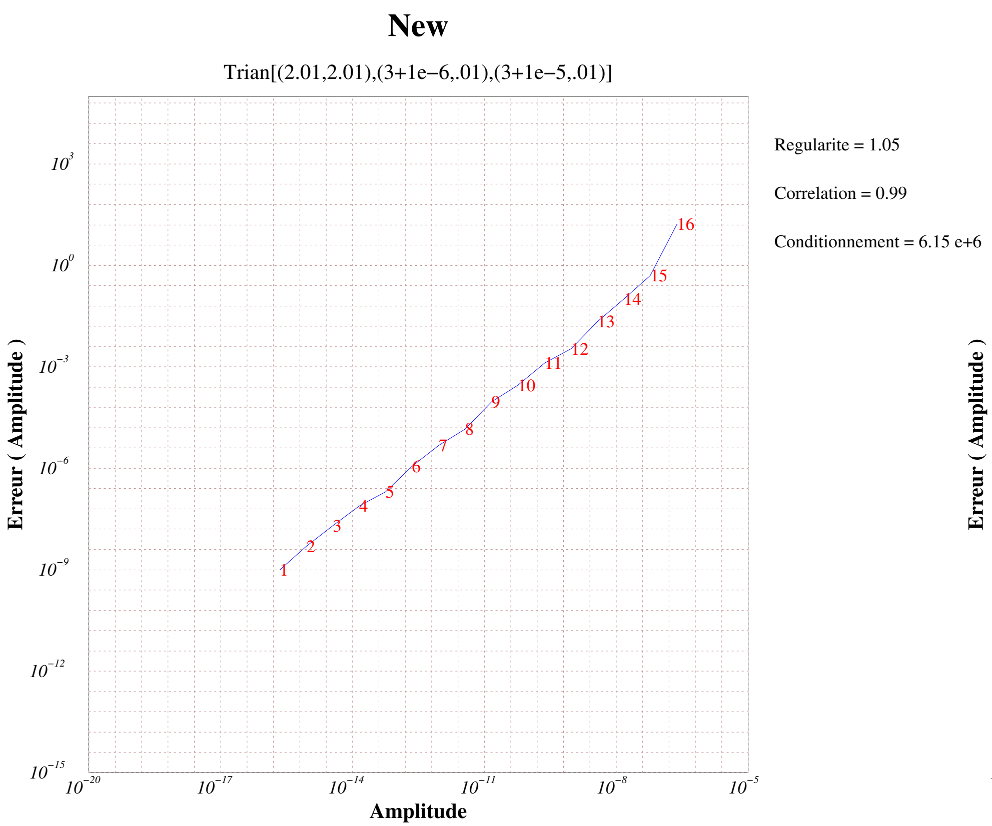
exemple 2

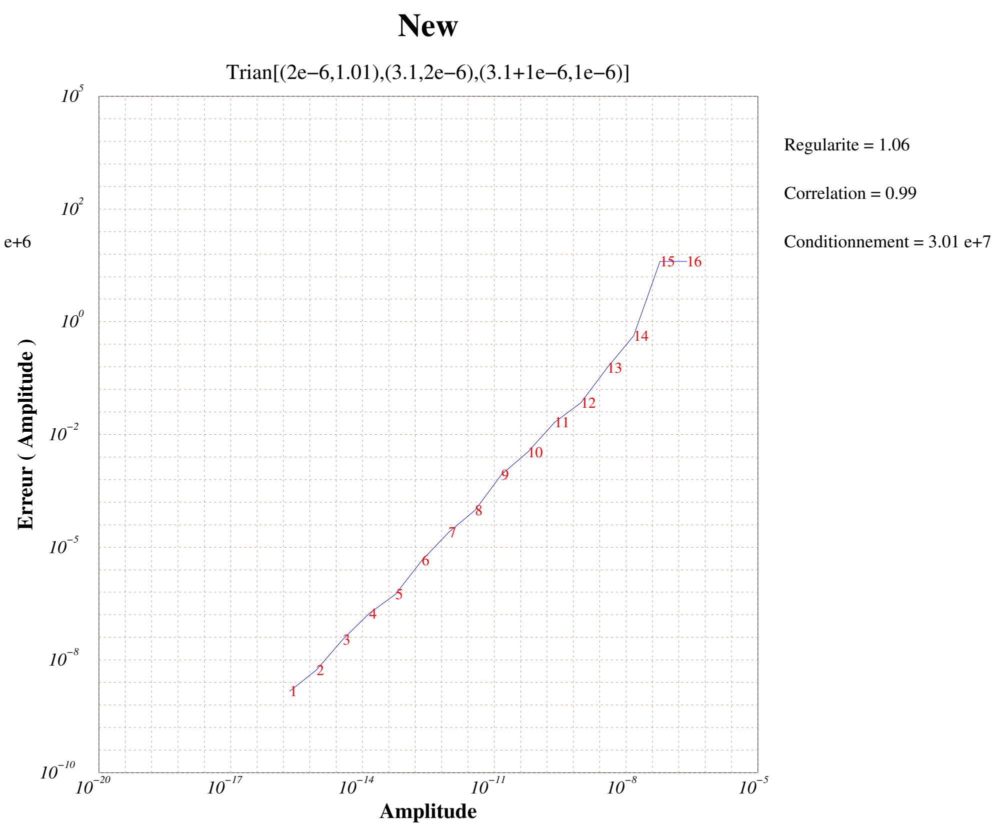


Le calcul est légèrement instable dans l'exemple 2

3^e cas : résultats avec le nouveau code - (2)

exemple 3 | exemple 4





Le calcul est stable dans les 2 exemples

Conclusion

Deux formules équivalentes dans $\mathbb{R}$
ont des comportements numériques différents

Une cancellation est souvent à l'origine des instabilités

Il faut faire une analyse mathématique et une étude de stabilité numérique
du logiciel pour garantir la précision du résultat

L'arithmétique d'intervalles,
l'analyse par perturbations des données,
l'arithmétique stochastique,
l'arithmétique multi-précision
sont des outils d'aide à l'étude de stabilité numérique