ResolSysteme [fr]

Des outils pour des systèmes linéaires, avec xint ou pyluatex.

Version 0.1.1 -- 7 Février 2023

Cédric Pierquet
c pierquet -- at -- outlook . fr
https://github.com/cpierquet/ResolSysteme

- Des commandes pour travailler sur des matrices carrées (2x2, 3x3 ou 4x4).
- Des commandes pour résoudre des systèmes linéaires (2x2, 3x3 ou 4x4).

Le **déterminant** de $A = \begin{pmatrix} -1 & 0,5 \\ \frac{1}{2} & 4 \end{pmatrix}$ est $\det(A) = -4{,}25$.

L'**inverse** de la matrice $A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 0 \\ 1 & 1 & 1 & 1 \\ -2 & -3 & -5 & -6 \end{pmatrix}$ est $A^{-1} = \begin{pmatrix} -15/8 & -1/8 & 3/2 & -1 \\ 23/8 & 1/8 & 1/2 & 2 \\ -9/8 & 1/8 & -3/2 & -1 \\ 1/8 & -1/8 & 1/2 & 0 \end{pmatrix}$.

La **solution** de $\begin{cases} y+z+t=1 \\ x+z+t=-1 \\ x+y+t=1 \\ x+y+z=0 \end{cases}$ est $\mathscr{S} = \left\{\left(-\frac{2}{3};\frac{4}{3};-\frac{2}{3};\frac{1}{3}\right)\right\}$.

LaTeX pdfLaTeX LuaLaTeX TikZ TeXLive MiKTeX

Table des matières

Première partie
Introduction

1 Le package ResolSysteme

1.1 Introduction

L'idée est de *proposer* des outils pour travailler sur des systèmes linéaires (de taille réduite !) :

— en affichant la **solution** (si elle existe) ;

— en affichant le **déterminant** et l'éventuelle **inverse** de la matrice des coefficients.

À noter que les calculs – en interne – peuvent être effectués de deux manières :

— via les packages xint* pour des formats **2x2** ou **3x3** ;

— via python et le package pyluatex (à charger manuellement du fait des options spécifiques) pour des formats **2x2**, **3x3** ou **4x4**.

Il n'est pas prévu – pour le moment – de travailler sur des matrices/systèmes plus grands.

L'utilisation de pyluatex nécessite une compilation adaptée, à savoir en LuaLaTeX et en activant le mode –shell-escape.
La méthode par python utilise quoi qu'il en soit le module `sympy`, qui doit donc être installé !

1.2 Packages utilisés, choix de formatage

Le package charge les packages suivantes :

— `xintexpr` et `xinttools` ;

— `sinuitx`, `nicefrac` et `nicematrix` ;

— `xstring` et `listofitems`.

Il est compatible avec les compilations usuelles en latex, pdflatex, lualatex (obligatoire pour pyluatex !!) ou xelatex.

Les nombres sont formatés par la commande `\num` de sinuitx, donc les options choisies par l'utilisateur se propageront aux résultats numériques.
L'affichage des matrices est gérée par le package nicematrix, et des options spécifiques *simples* pourront être placées dans les différentes commandes.

1.3 Chargement du package, et option

Le package peut donc se charger de deux manières différentes, suivant si l'utilisateur utilise python ou non. Les commandes *classiques* sont disponibles même si python est utilisé.

Code LaTeX

```
%chargement du package sans passer par pyluatex, calculs via xint
\usepackage{ResolSysteme}
```

Code LaTeX

```
%chargement du package pyluatex et du package avec [pyluatex]
\usepackage[options]{pyluatex}
\usepackage[pyluatex]{ResolSysteme}
```

Deuxième partie
Commandes

2 Une commande interne : écriture sous forme d'une fraction

2.1 La commande

En *interne*, le code utilise une commande pour formater un résultat sous forme fractionnaire, avec gestion des entiers et gestion du signe « – ».

Code LaTeX
```
\ConvVersFrac(*)[option de formatage]{calcul}
```

2.2 Utilisation

Concernant cette commande, qui est dans un bloc `ensuremath` :

- la version *étoilée* force l'écriture du signe « – » avant l'éventuelle fraction ;
- le premier argument, *optionnel* et entre [...] permet de spécifier un formatage du résultat :
  - ⟨**t**⟩ pour l'affichage de la fraction en mode tfrac ;
  - ⟨**d**⟩ pour l'affichage de la fraction en mode dfrac ;
  - ⟨**n**⟩ pour l'affichage de la fraction en mode nicefrac ;
  - ⟨**dec**⟩ pour l'affichage du résultat en mode `décimal` (sans arrondi !) ;
  - ⟨**dec=k**⟩ pour l'affichage du résultat en mode `décimal` arrondi à 10^{-k} ;
- le second argument, *obligatoire*, est quant à lui, le calcul en syntaxe xint.

Code LaTeX
```
\ConvVersFrac{-10+1/3*(-5/16)}            %sortie par défaut (fraction avec - sur numérateur)

\ConvVersFrac*{-10+1/3*(-5/16)}           %sortie avec - avant la fraction

\ConvVersFrac*[d]{-10+1/3*(-5/16)}        %sortie en displaystyle

\ConvVersFrac[n]{-10+1/3*(-5/16)}         %sortie en nicefrac

\ConvVersFrac[dec=4]{-10+1/3*(-5/16)}     %sortie en décimal arrondi à 0,0001

\ConvVersFrac{2+91/7}                     %entier correctement formaté
```

Code LaTeX

$\frac{-485}{48}$ $-\frac{485}{48}$ $-\dfrac{485}{48}$ $-485/48$ $-10{,}1042$ 15

2.3 Interaction avec les commandes « matricielles », limitations

En *interne*, le formatage des résultats est géré par cette commande, et les options disponibles existent donc de la même manière pour les commandes liées aux systèmes linéaires.

Il ne sera par contre pas possible de spécifier des options différentes pour chacun des coefficients, autrement dit l'éventuelle option se propagera sur l'ensemble des résultats !

Les *transformations* en fraction devraient pouvoir fonctionner avec des calculs *classiques*, mais il est possible que, dans des cas *spécifiques*, les résultats ne soient pas ceux attendus !

3 Calcul de déterminant

3.1 Introduction

La première commande (matricielle) disponible est pour calculer le déterminant d'une matrice :

— **2x2** ou **3x3** pour le package *classique* ;

— **2x2** ou **3x3** ou **4x4** pour le package *lua*.

Code LaTeX

```
%version classique
\DetMatrice(*)[option de formatage](matrice)

%version python
\DetMatricePY(*)[option de formatage](matrice)
```

3.2 Utilisation

Concernant cette commande, qui est à insérer dans un environnement *math* :

— la version *étoilée* force l'écriture du signe « − » avant l'éventuelle fraction ;

— le premier argument, *optionnel* et entre [...] permet de spécifier un formatage du résultat :

  — ⟨**t**⟩ pour l'affichage de la fraction en mode tfrac ;

  — ⟨**d**⟩ pour l'affichage de la fraction en mode dfrac ;

  — ⟨**n**⟩ pour l'affichage de la fraction en mode nicefrac ;

  — ⟨**dec**⟩ pour l'affichage du résultat en mode `décimal` (sans arrondi !) ;

  — ⟨**dec=k**⟩ pour l'affichage du résultat en mode `décimal` arrondi à 10^{-k} ;

— le second argument, *obligatoire* et entre (...), est quant à lui, la matrice donnée par ses coefficients a11,a12,... § a21,a22,... (syntaxe *héritée* de sympy).

Code LaTeX

```
%version classique
Le dét. de $A=\begin{pNiceMatrix}1&2\\3&4\end{pNiceMatrix}$ est
$\det(A)=\DetMatrice(1,2 § 3,4)$.
```

Le dét. de $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ est $\det(A) = -2$.

Code LaTeX

```
%version classique
Le dét. de $A=\begin{pNiceMatrix}-1&{0,5}\\\frac12&4\end{pNiceMatrix}$ est
$\det(A)=\DetMatrice[dec](-1,0.5 § 1/2,4)$.
```

Le dét. de $A = \begin{pmatrix} -1 & 0,5 \\ \frac{1}{2} & 4 \end{pmatrix}$ est $\det(A) = -4{,}25$.

Code LaTeX

```
%version classique
Le dét. de $A=\begin{pNiceMatrix}-1&\frac13&4\\\frac13&4&-1\\-1&0&0\end{pNiceMatrix}$ est
$\det(A) \approx \DetMatrice[dec=3](-1,1/3,4 § 1/3,4,-1 § -1,0,0)$.
```

Le dét. de A $= \begin{pmatrix} -1 & \frac13 & 4 \\ \frac13 & 4 & -1 \\ -1 & 0 & 0 \end{pmatrix}$ est $\det(A) \approx 16{,}333$.

Code LaTeX

```
%version python
Le dé. de $A=\begin{pNiceMatrix}1&2\\3&4\end{pNiceMatrix}$ est
$\det(A)=\DetMatricePY(1,2 § 3,4)$.
```

Le dé. de A $= \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ est $\det(A) = -2$.

Code LaTeX

```
Le dét. de $A=\begin{pNiceMatrix}-1&{0,5}\\\frac12&4\end{pNiceMatrix}$ est
$\det(A)=\DetMatricePY*[d](-1,0.5 § 1/2,4)$.
```

Le dét. de A $= \begin{pmatrix} -1 & 0,5 \\ \frac12 & 4 \end{pmatrix}$ est $\det(A) = -\dfrac{17}{4}$.

Code LaTeX

```
%version python
Le dét. de $A=\begin{pNiceMatrix}-1&\frac13&4\\\frac13&4&-1\\-1&0&0\end{pNiceMatrix}$ est
$\det(A) \approx \DetMatricePY[dec=3](-1,1/3,4 § 1/3,4,-1 § -1,0,0)$.
```

Le dét. de A $= \begin{pmatrix} -1 & \frac13 & 4 \\ \frac13 & 4 & -1 \\ -1 & 0 & 0 \end{pmatrix}$ est $\det(A) \approx 16{,}333$.

4 Inverse d'une matrice

4.1 Introduction

La deuxième commande (matricielle) disponible est pour calculer l'éventuelle inverse d'une matrice :

— **2x2** ou **3x3** pour le package *classique* ;

— **2x2** ou **3x3** ou **4x4** pour le package *lua*.

Code LaTeX

```
%version classique
\MatriceInverse(*)[option de formatage]<options nicematrix>(matrice)

%version python
\MatriceInversePY(*)[option de formatage]<options nicematrix>(matrice)
```

4.2 Utilisation

Concernant cette commande, qui est à insérer dans un environnement *math* :

- la version *étoilée* force l'écriture du signe « – » avant l'éventuelle fraction ;
- le premier argument, *optionnel* et entre [...] permet de spécifier un formatage du résultat :
  - ⟨**t**⟩ pour l'affichage de la fraction en mode tfrac ;
  - ⟨**d**⟩ pour l'affichage de la fraction en mode dfrac ;
  - ⟨**n**⟩ pour l'affichage de la fraction en mode nicefrac ;
  - ⟨**dec**⟩ pour l'affichage du résultat en mode `décimal` (sans arrondi !) ;
  - ⟨**dec=k**⟩ pour l'affichage du résultat en mode `décimal` arrondi à 10^{-k} ;
- le deuxième argument, *optionnel* et entre <...> correspond aux ⟨**options**⟩ à passer à l'environnement `pNiceMatrix` ;
- le troisième argument, *obligatoire* et entre (...), est quant à lui, la matrice donnée par ses coefficients a11,a12,... § a21,a22,... (syntaxe héritée de `sympy`).

À noter que si la matrice n'est pas inversible, le texte `Matrice non inversible` est affiché.

Code LaTeX

```
%version classique
L'inverse de $A=\begin{pNiceMatrix}1&2\\3&4\end{pNiceMatrix}$ est
$A^{-1}=\MatriceInverse<cell-space-limits=2pt>(1,2 § 3,4)$.
```

L'inverse de $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ est $A^{-1} = \begin{pmatrix} -2 & 1 \\ \frac{3}{2} & \frac{-1}{2} \end{pmatrix}$.

Code LaTeX

```
%version classique
L'inverse de $A=\begin{pNiceMatrix}1&2&3\\4&5&6\\7&8&8\end{pNiceMatrix}$ est
$A^{-1}=\MatriceInverse[n]<cell-space-limits=2pt>(1,2,3 § 4,5,6 § 7,8,8)$.
```

L'inverse de $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 8 \end{pmatrix}$ est $A^{-1} = \begin{pmatrix} -8/3 & 8/3 & -1 \\ 10/3 & -13/3 & 2 \\ -1 & 2 & -1 \end{pmatrix}$.

Code LaTeX

```
%version python
L'inverse de $A=\begin{pNiceMatrix}1&2\\3&4\end{pNiceMatrix}$ est
$A^{-1}=\MatriceInversePY*[d]<cell-space-limits=2pt>(1,2 § 3,4)$.
```

L'inverse de $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ est $A^{-1} = \begin{pmatrix} -2 & 1 \\ \dfrac{3}{2} & -\dfrac{1}{2} \end{pmatrix}$.

```
%version python
L'inv. de $A=\begin{pNiceMatrix}1&2&3&4\\5&6&7&0\\1&1&1&1\\-2&-3&-5&-6\end{pNiceMatrix}$ est
$A^{-1}=
\MatriceInversePY*[n]<cell-space-limits=2pt>(1,2,3,4 § 5,6,7,0 § 1,1,1,1 § -2,-3,-5,-6)$.
```

L'inv. de $A=\begin{pmatrix}1&2&3&4\\5&6&7&0\\1&1&1&1\\-2&-3&-5&-6\end{pmatrix}$ est $A^{-1}=\begin{pmatrix}-\nicefrac{15}{8}&-\nicefrac{1}{8}&\nicefrac{3}{2}&-1\\\nicefrac{23}{8}&\nicefrac{1}{8}&\nicefrac{1}{2}&2\\-\nicefrac{9}{8}&\nicefrac{1}{8}&-\nicefrac{3}{2}&-1\\\nicefrac{1}{8}&-\nicefrac{1}{8}&\nicefrac{1}{2}&0\end{pmatrix}$.

5 Résolution d'un système linéaire

5.1 Introduction

La deuxième commande (matricielle) disponible est pour déterminer l'éventuelle solution d'un système linéaire qui s'écrit matriciellement $A \times X = B$:

— **2x2** ou **3x3** pour le package *classique* ;

— **2x2** ou **3x3** ou **4x4** pour le package *lua*.

```
%version classique
\SolutionSysteme(*)[opt de formatage]<opts nicematrix>(matriceA)(matriceB)[opt Matrice]

%version python
\SolutionSystemePY(*)[opt de formatage]<opts nicematrix>(matriceA)(matriceB)[opt Matrice]
```

5.2 Utilisation

Concernant cette commande, qui est à insérer dans un environnement *math* :

— la version *étoilée* force l'écriture du signe « − » avant l'éventuelle fraction ;

— le premier argument, *optionnel* et entre [...] permet de spécifier un formatage du résultat :

  — ⟨**t**⟩ pour l'affichage de la fraction en mode tfrac ;

  — ⟨**d**⟩ pour l'affichage de la fraction en mode dfrac ;

  — ⟨**n**⟩ pour l'affichage de la fraction en mode nicefrac ;

  — ⟨**dec**⟩ pour l'affichage du résultat en mode `décimal` (sans arrondi !) ;

  — ⟨**dec=k**⟩ pour l'affichage du résultat en mode `décimal` arrondi à 10^{-k} ;

— le deuxième argument, *optionnel* et entre <...> correspond aux ⟨**options**⟩ à passer à l'environnement `pNiceMatrix` ;

— le troisième argument, *obligatoire* et entre (...), est quant à lui, la matrice A donnée par ses coefficients a11,a12,... § a21,a22,... (syntaxe héritée de `sympy`) ;

— le quatrième argument, *obligatoire* et entre (...), est quant à lui, la matrice B donnée par ses coefficients b11,b21,... (syntaxe héritée de `sympy`) ;

— le dernier argument, *optionnel* et entre [...], permet – grâce à ⟨**Matrice**⟩ – de présenter le vecteur solution.

À noter que si la matrice n'est pas inversible, le texte `Matrice non inversible` est affiché.

Code LaTeX

```
%version classique
La solution de $\systeme{3x+y-2z=-1,2x-y+z=4,x-y-2z=5}$ est $\mathcal{S}=%
\left\lbrace \SolutionSysteme*[d](3,1,-2 § 2,-1,1 § 1,-1,-2)(-1,4,5) \right\rbrace$.\\
```

La solution de $\begin{cases} 3x+y-2z=-1 \\ 2x-y+z=4 \\ x-y-2z=5 \end{cases}$ est $\mathcal{S}=\left\lbrace \left(\frac{1}{2};-\frac{7}{2};-\frac{1}{2}\right) \right\rbrace$.

Code LaTeX

```
%version python
La solution de $\systeme{x+y+z=-1,3x+2y-z=6,-x-y+2z=-5}$ est $\mathcal{S}=%
\left\lbrace \SolutionSystemePY(1,1,1 § 3,2,-1 § -1,-1,2)(-1,6,-5) \right\rbrace$.
```

La solution de $\begin{cases} x+y+z=-1 \\ 3x+2y-z=6 \\ -x-y+2z=-5 \end{cases}$ est $\mathcal{S}=\{(2;-1;-2)\}$.

Code LaTeX

```
%version python
La solution de $\systeme[xyzt]{x+2y+3z+4t=-10,5x+6y+7z=0,x+y+z+t=4,-2x-3y-5z-6t=7}$
est $\mathcal{S}=%
\left\lbrace
    \SolutionSystemePY%
        [dec]<cell-space-limits=2pt>%
        (1,2,3,4 § 5,6,7,0 § 1,1,1,1 § -2,-3,-5,-6)(-10,0,4,7)%
        [Matrice]
\right\rbrace$.
```

La solution de $\begin{cases} x+2y+3z+4t=-10 \\ 5x+6y+7z=0 \\ x+y+z+t=4 \\ -2x-3y-5z-6t=7 \end{cases}$ est $\mathcal{S}=\left\lbrace \begin{pmatrix} 17{,}75 \\ -12{,}75 \\ -1{,}75 \\ 0{,}75 \end{pmatrix} \right\rbrace$.

Code LaTeX

```
%pas de solution
La solution de $\systeme{x+2y=-5,4x+8y=1}$ est $\mathcal{S}=%
\left\lbrace \SolutionSystemePY(1,2 § 4,8)(-5,1) \right\rbrace$.
```

La solution de $\begin{cases} x+2y=-5 \\ 4x+8y=1 \end{cases}$ est $\mathcal{S}=\{\text{Matrice non inversible}\}$.

Troisième partie

Fonctions python utilisées

Les fonctions utilisées par les packages pyluatex ou pythontex sont données ci-dessous.
Elles sont accessibles en *natif* une fois l'option lua activée, grâce notamment à la macro \py.

Code Python

```python
import sympy as sy
x = sy.Symbol('x')
y = sy.Symbol('y')
z = sy.Symbol('z')
t = sy.Symbol('t')

def resol_systeme_QQ(a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,u) :
  solution=sy.solve([a*x+b*y+c*z+d*t-e,f*x+g*y+h*z+i*t-j,k*x+l*y+m*z+n*t-o,p*x+q*y+r*z+s*t-u],[x,y,z,t])
  return solution

def resol_systeme_TT(a,b,c,d,e,f,g,h,i,j,k,l) :
  solution=sy.solve([a*x+b*y+c*z-d,e*x+f*y+g*z-h,i*x+j*y+k*z-l],[x,y,z])
  return solution

def resol_systeme_DD(a,b,c,d,e,f) :
  solution=sy.solve([a*x+b*y-c,d*x+e*y-f],[x,y])
  return solution

def det_matrice_QQ(a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p) :
  MatTmp = sy.Matrix(([a,b,c,d],[e,f,g,h],[i,j,k,l],[m,n,o,p]))
  DetMatTmp = MatTmp.det()
  return DetMatTmp

def det_matrice_TT(a,b,c,d,e,f,g,h,i) :
  MatTmp = sy.Matrix(([a,b,c],[d,e,f],[g,h,i]))
  DetMatTmp = MatTmp.det()
  return DetMatTmp

def det_matrice_DD(a,b,c,d) :
  MatTmp = sy.Matrix(([a,b],[c,d]))
  DetMatTmp = MatTmp.det()
  return DetMatTmp

def inverse_matrice_QQ(a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p) :
  MatTmp = sy.Matrix(([a,b,c,d],[e,f,g,h],[i,j,k,l],[m,n,o,p]))
  DetMatTmp = MatTmp.inv()
  return DetMatTmp

def inverse_matrice_DD(a,b,c,d) :
  MatTmp = sy.Matrix(([a,b],[c,d]))
  InvMatTmp = MatTmp.inv()
  return InvMatTmp

def inverse_matrice_TT(a,b,c,d,e,f,g,h,i) :
  MatTmp = sy.Matrix(([a,b,c],[d,e,f],[g,h,i]))
  InvMatTmp = MatTmp.inv()
  return InvMatTmp
```

Quatrième partie
Historique

`v0.1.1` : Correction d'un bug avec le caractère « ; »
`v0.1.0` : Version initiale