Analyse de pracma par Tarik Hakam

Wenjun ZHAO

12/20/2020

I. Introduction

Vous pouvez retrouver mon travail sur mon Github, **https://github.com/claudia0524/PSBX**.

Pour ce dossier, nous avons étudier le travail de Tarik Hakam, étudiant en M2DM au sein de PSB.

Nous avons trouvé son dossier sur son Github : **https://github.com/T-Hak/PSBX**

Maintenant, nous allons établir mes 5 critères d'évaluations, qui seront les mêmes pour tout les dossiers que j'ai vais étudier pour ce devoir:

Les cinq critères d'évaluation

1. Rmd se comporte bien à l'exécution

2. Les aspects intéressant, didactique, complet

3. La qualité Rmarkdown, la qualité de l'écriture

4. Didactique, conformité aux exigences vues plus haut et comporte du calcul symbolique et du calcul numérique

5. La qualité du LaTeX et des illustrations, la qualité de l'écriture, le choix des ressource internet, la compréhension personnelle des concepts

II. Synthèse du travail en question

Pracma est fournit un grand nombre de fonctions d'analyse numérique et d'algèbre linéaire, d'optimisation numérique, d'équations différentielles, de séries chronologiques, ainsi que de certaines fonctions mathématiques spéciales bien connues. Utilise les noms de fonction 'MATLAB' le cas échéant pour simplifier le portage.

Nous allons maintenant étudier le code de Tarik dans le chapitre suivant.

III. Extrait commenté des parties de code

Cas d'usage du package pracma : Quadrature Hermite-Gauss

La quadrature Hermite-Gauss, également appelée quadrature Hermite, est une quadrature gaussienne sur l'intervalle $(-\infty, \infty)$ avec fonction de pondération w telle que (**???**):

$$w(x) = \exp(-x^2) \tag{1}$$

La quadrature Hermite est utilisée pour intégrer des fonctions de la forme(Tarik, n.d.):

$$\int_{-\infty}^{\infty} f(x) \exp(-x^2)\, dx \tag{2}$$

x et w sont obtenus à partir des valeurs propres tridiagonales.

La valeur d'une telle intégrale est alors sum($w \times f(x)$). (**???**)

R Code

Pour coder la quadrature Hermite-Gauss, le package pracma a une fonction prédéfinie : **gaussHermite(n)**

Dans notre exemple, nous prenons $n = 17$ comme ordre de quadrature.

```r
#Gauss-Hermite Quadrature Formula
#gaussHermite(n) avec ici n = 17
library("pracma")
f <- gaussHermite(17)
```

Ensuite, nous calculons l'intégrale suivante :

$$\int_{-\infty}^{\infty} \exp(-x^2)\, dx \tag{3}$$

Comme vu dans la section précédente, *"La valeur d'une telle intégrale est alors sum($w \times f(x)$)."*

```r
sum(f$w) #=> 1.77245385090552 == sqrt(pi)
```

```
## [1] 1.772454
```

Ainsi, nous obtenons la résolution suivante :

$$\int_{-\infty}^{\infty} \exp(-x^2)\, dx = \sqrt{\pi} \tag{4}$$

Nous calculons de la même façon les intégrales suivantes :

$$\int_{-\infty}^{\infty} x^2 \exp(-x^2)\, dx = \frac{\sqrt{\pi}}{2} \tag{5}$$

$$\int_{-\infty}^{\infty} cos(x) \exp(-x^2)\, dx = \frac{\sqrt{\pi}}{\exp(1)^{1/4}} \tag{6}$$

Pour trouver ces solutions, nous avons codé les commandes suivantes :

```r
# Integrate x^2 exp(-x^2)
sum(f$w * f$x^2) #=> 0.886226925452076 == sqrt(pi)/2
```

```
## [1] 0.8862269
```

```r
# Integrate cos(x) * exp(-x^2)
sum(f$w * cos(f$x)) #=> 1.38038844704314 == sqrt(pi)/exp(1)^0.25
```

```
## [1] 1.380388
```

To dig depper

Les abscisses pour l'ordre de quadrature n sont données par les racines x_i des polynômes Hermite $H_n(x)$, dont les points sont positionnés symétriquement autour de 0. Les poids sont :

$$w_i = -\frac{A_{n+1}\gamma_n}{A_n H_n^{(x_i)} H'_{n+1}(x_i)} = \frac{A_n}{A_{n-1}} \frac{\gamma_{n-1}}{H_{n-1}(x_i) H'_n(x_i)} \tag{7}$$

où,

A_n est le coefficient de x^n dans $H_n(x)$.

Pour les polynômes Hermite,

$$A_n = 2^n \Leftrightarrow \frac{A_{n+1}}{A_n} = 2 \tag{8}$$

De plus,

$$\gamma_n = \sqrt(\pi) 2^n n!$$

$$w_i = -\frac{2^{n+1} n! \sqrt(\pi)}{H_{n+1}(x_i) H'_n(x_i)} = \frac{2^n (n-1)! \sqrt(\pi)}{H_{n-1}(x_i) H'_n(x_i)} = \frac{2^{n+1} n! \sqrt(\pi)}{[H'_n(x_i))]^2} = \frac{2^{n+1} n! \sqrt(\pi)}{[H_{n+1}(x_i)]^2} = \frac{2^{n-1} n! \sqrt(\pi)}{n^2 [H_{n-1}(x_i)]^2} \tag{9}$$

en utilisant la relation de récurrence suivante :

$$H'_n(x) = 2n H_{n-1}(x) = 2x H_n(x) - H_{n+1}(x) \tag{10}$$

ainsi, on obtient :

$$H'_n(x_i) = 2n H_{n-1}(x_i) = -H_{n+1}(x_i) \tag{11}$$

Cas d'usage du package pracma : Approximation de Tchebytchev

Cette méthode traite des polynômes uniquement défini sur l'intervalle [-1;1] tandis que la fonction à étudier est sur l'intervalle [a;b] avec $a, b \in \mathbf{R}$.

Toute variable x de ce dernier correspondra une variable v comprise entre -1 et 1, par transformation affine suivante :

$$v = \frac{2(x-a)}{(b-a)-1} \tag{12}$$

L'idée de Tchebychev est d'approcher le calcul de f(x) par la formule :

$$f(x) = \sum_{k=0}^{n} A_k \times T_k(v) \tag{13}$$

$T_k(v)$ étant un polynôme de Tchebychev de degré k en v, ainsi défini :

$$T_k(v) = 2 \times v \times T_{[k-1]}(v) - T_{[k-2]}(v) \tag{14}$$

Code R

Nous allons approximer $sin(x)$ avec l'approximation de Tchebytchev sur $[-\pi, \pi]$. Pour commencer, nous avons comparer $sin(x)$ avec un polynôme de degré $n = 9$. On parle d'évaluation polynomiale.

L'exemple est le suivant :

$$P(x) = x - \frac{1}{6}x^3 + \frac{1}{120}x^5 - \frac{1}{5040}x^7 + \frac{1}{362880}x^9 \tag{15}$$

Ainsi, on code un polynôme de la façon suivante :

```
#Approximate sin(x) on [-pi, pi] with a polynomial of degree 9
# Polynomial:
p1 <- rev(c(0, 1, 0, -1/6, 0, 1/120, 0, -1/5040, 0, 1/362880))
```

Ensuite pour l'approximation de Tchebytchev, on définit les coefficients de Tchebytchev pour les polynômes de Tchebytchev.

```
#Compare
#chebCoeff(f, a, b, n)
p2 <- chebCoeff(sin, -pi, pi, 9)
```

On découpe ensuite l'intervalle $[-\pi, \pi]$, ici en 101 chuncks. Plus le pas est petit, plus l'approximation est "juste".

```
# Estimate the maximal distance
x <- seq(-pi, pi, length.out = 101)
#length.out = length of the sequence
```

Afin de comparer l'efficacité de l'approximation de Tchebytchev et celle de l'évaluation polynomiale, on définit les fonctions suivantes : la fonction à approximer $sin(x)$, l'évaluation polynomiale yp, l'approximation de Tchebytchev yc.

```
ys <- sin(x)
yp <- polyval(p1, x)
yc <- chebApprox(x, sin, -pi, pi, 9)
```

On obtient les écarts suivants :

```
max(abs(ys-yp)) # 0.006925271
```

```
## [1] 0.006925271
```

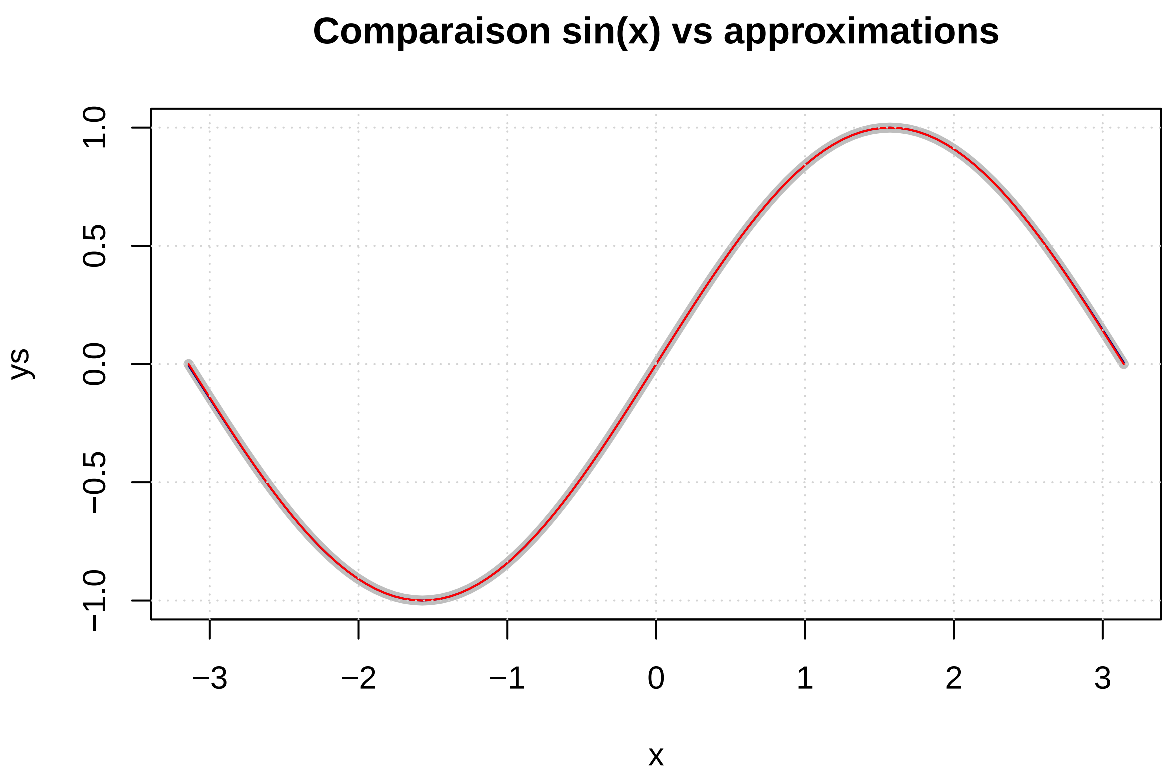
```
max(abs(ys-yc)) # 1.151207e-05
```

```
## [1] 1.151207e-05
```

On peut ainsi conclure que l'approximation de Tchebytchev est plus efficace que l'évaluation polynomiale.

Pour finir, on "plot" les fonctions de l'exemple pour illustrer.

```
# Plot the corresponding curves
plot(x, ys, type = "l", col = "gray", lwd = 5,
     main="Comparaison sin(x) vs approximations")
lines(x, yp, col = "navy")
lines(x, yc, col = "red")
grid()
```



IV. Evaluation du travail en question

1. Rmd se comporte bien à l'exécution (4/4)

2. Les aspects intéressant, didactique, complet (3/4)

3. La qualité Rmarkdown, la qualité de l'écriture (4/4)

4. Didactique, conformité aux exigences vues plus haut et comporte du calcul symbolique et du calcul numérique (3/4)

5. La qualité du LaTeX et des illustrations, la qualité de l'écriture, le choix des ressource internet, la compréhension personnelle des concepts (3/4)

V. Conclusion

En général, ce tracail de Tarik exécuse bien dans l'environement de R. Et il nous explique clairement comment fonctionner le package GGPLOT2 dans un rmd. L'aspect est intéressant, complet et propre. La qualité Rmarkdown et la qualité de l'écriture sont ainsi bien. Didactique, conformité aux exigences vues plus haut. La qualité du LaTeX fonctionne bien aussi, et il peut montrer le but principal de package de ggplot2. Le mieux point est qu'il présente des ressources internet. Il utilise des calcul symbolique et du calcul numérique.

VI. Bibliographie

Tarik, Hakam. n.d. "Creation Bibliography." https://github.com/T-Hak/PSBX.