Université Paris I – Panthéon-Sorbonne
90, rue de Tolbiac – 75634 Paris Cedex 13

Masters MAEF, QEM, DU MMEF — June 2016
Resit exam

Introduction to Modern C++

(English version)

No documents, calculators, or computers allowed
Duration: 2h00.

Note: Please write clear and concise answers. Make sure that punctuation, if any, is visible. Most questions can be treated independently. Feel free to use diagrams and/or colour if it makes your answers clearer.

We will consider throughout this exam the *stochastic neoclassical growth model*, which is the foundation of much work in macroeconomics. In this model, a social planner picks a sequence of consumption c_t and capital k_t, so as to solve

$$\max_{\{c_t, k_{t+1}\}} \mathbb{E}_0 \left[\sum_{t=0}^{\infty} (1-\beta)\beta^t \log c_t \right]$$

where $\mathbb{E}_0$ is the conditional expectation operation, and β is the discount factor. Of course, resources are constrained, which is modeled here as:

$$c_t + k_{t+1} = z_t k_t^\alpha$$

where productivity z_t takes values in a set of points $\{z_1, \dots, z_n\}$ that evolve according to a Markov transition matrix.

One of the most efficient ways to solve this problem is by solving the corresponding Bellman equation. We thus introduce

$$V(k,z) = \max_{k'} (1-\beta)\beta^t \log(zk^\alpha - k') + \beta \mathbb{E}\left[V(k',z')|z\right]$$

Note that V appears in the left-hand side as well as the right-hand side, which makes this problem mathematically challenging. But it is possible to solve this equation iteratively, introducing

$$V^i(k,z) = \max_{k'} (1-\beta)\beta^t \log(zk^\alpha - k') + \beta \mathbb{E}\left[V^{i-1}(k',z')|z\right]$$

Then, starting from any initial V^0 we compute V^1, V^2, etc. and standard arguments show that in fact $V^i \xrightarrow{i\to\infty} V$ (in the sup norm).

While we won't have time to go all the way through the full resolution of this problem, the goal of this exam is to solve some parts of it.

Question 1. We choose $\alpha = 1/3$, $\beta = 0.95$. Define the two corresponding C++ constants with appropriate name and type.

Question 2. We choose $\{z_1, \dots, z_n\} = \{0.9792, 0.9896, 1.0000, 1.0106, 1.0212\}$. Define a C++ `vector` (not an array), called `z`, that contains these values. What standard library should we include to use `vectors`?

Question 3. We also define the three following intermediary variables:

```
double capitalSteadyState     = std::pow(alpha*beta,1./(1-alpha));
double outputSteadyState      = std::pow(capitalSteadyState,alpha);
double consumptionSteadyState = outputSteadyState-capitalSteadyState;
```

What is (mathematically) the value of these variables? What is the difference between `1` and `1.` ?

Question 4. In Question 3's code, what problem could happen? Give all the values of α and β that could cause a crash.

Question 5. Write an `if-then-else` clause that captures bad values of α or β and displays an error message if appropriate.

Question 6. If `alpha` and `beta` belonged to a C++ class, they could have `private`, `protected`, or `public` scope. Explain what each of these means.

Question 7. To solve the problem, we construct a "grid" `gridCapital` of size `17820`, such that

```
gridCapital[i] = 0.5 * capitalSteadyState + 0.00001 * i;
```

Write a `for loop` that initialises `gridCapital`. What could be the type of `gridCapital`? Bonus question: What could be the type of i?

Question 8. Create a function `InitialiseGrid` that takes `gridCapital` as an input, and populates it as in Question 7. Make sure to use the appropriate modifiers in the argument list (e.g. `const`, `&`, `&&`, `[]`, `static`, ...). Is `InitialiseGrid` a pure or an impure function, and why?

Question 9. We will now work with matrices. C++ does not have a native matrix type, but we can make one as follows:

```
template <std::size_t Rows, std::size_t Columns> using Matrix =
    std::array<std::array<double, Columns>, Rows>;
```

Our matrices will be $N \times M$, where $N = 17820$ and $M = 5$. Write the signature of a function `InitialiseExpectedValue` that takes no argument and returns a `Matrix` of the appropriate size. We are not interested here in how this function works (it is very simple though).

Question 10. Write a `const`-safe function `CopyMatrix` that takes as input a matrix A and returns a new matrix that is a copy of A. What is the interest of `const`-safety?

Question 11. The complete program has the following form:

1. Definition of constants (Question 1)
2. Initialisation of the grid (Questions 7–8)
3. Repeat a certain number of times:
   (a) Initialisation of the expected value matrix (Question 9)
   (b) Computation of the Bellman equation V^i (not treated here)
   (c) Copy of the new matrix (Question 10)
4. The final result is the matrix that we computed in the previous step.

Assume we have written the whole program. We now have to compile it, and we use the following command line:

```
g++ -o rbc rbc.cpp -O3 --std=C++11
```

Explain every aspect of this command.

Question 12. We have put our code inside a function `OptimalBehaviour`, that takes as input α and β, and outputs the sequence $\{c_t, k_t\}$ for all $0 \leq t < T$. We wish to use that function in a larger program, what should we do?

☐ Write proper documentation to explain which values of α and β can be used

☐ Put the function inside a class of its own

☐ Put the function inside a namespace of its own

☐ Create a template with type definitions

☐ Create a header file and add the header file to the compilation chain

☐ Create a header file and add the code file to the compilation chain

☐ Create a header file and add both the code and header file to the compilation chain

Bonus question: Write a recursive, pure, type-safe, `const`-safe, documented function that computes the square root of x up to a given precision, using that $r_k \xrightarrow{k\to\infty} \sqrt{x}$ where $r_{k+1} = \frac{1}{2}(r_k + x/r_k)$.

Université Paris I – Panthéon-Sorbonne
90, rue de Tolbiac – 75634 Paris Cedex 13

Masters MAEF, QEM, DU MMEF — Juin 2016
Examen de rattrapage

Introduction au C++ moderne

(Version française)

Documents, calculateurs et ordinateurs interdits
Durée: 2h00.

Remarque: Merci de fournir des réponses claires et concises. Assurez-vous que la ponctuation, s'il y a lieu, est visible. La plupart des questions peuvent être traitées indépendamment des autres. Si utile, vous pouvez utiliser des diagrammes ou des couleurs pour rendre vos réponses plus claires.

Dans cet examen on s'intéresse au *modèle néoclassique stochastique de la croissance*, qui est à la base de beaucoup de travaux en macroéconomie. Dans ce modèle, on doit choisir à chaque instant une consommation c_t et un capital k_t, de sorte que

$$\max_{\{c_t,k_{t+1}\}} \mathbb{E}_0 \left[\sum_{t=0}^{\infty} (1-\beta)\beta^t \log c_t \right]$$

où $\mathbb{E}_0$ est l'espérance conditionelle, et β est un facteur de dépréciation. Les ressources sont limitées, ce que l'on modélise ici par:

$$c_t + k_{t+1} = z_t k_t^\alpha$$

avec un productivité z_t qui prend ses valeurs parmi $\{z_1, \ldots, z_n\}$ selon un échantillonnage de Markov.

Une des manières les plus efficaces de résoudre ce problème est d'utiliser la méthode de Bellman. On introduit donc

$$V(k,z) = \max_{k'} (1-\beta)\beta^t \log(zk^\alpha - k') + \beta \mathbb{E}\left[V(k',z')|z\right]$$

Notez que V apparaît à la gauche et à la droite de cette équation, ce qui la rend difficile à attaquer mathématiquement. Mais il est possible de la résoudre numériquement, en posant

$$V^i(k,z) = \max_{k'} (1-\beta)\beta^t \log(zk^\alpha - k') + \beta \mathbb{E}\left[V^{i-1}(k',z')|z\right]$$

En partant de n'importe quel V^0, on calcule alors V^1, V^2, etc. et par des arguments standards on montre que $V^i \xrightarrow{i\to\infty} V$ (pour la norme sup).

Nous n'aurons pas le temps de tout traiter de cette résolution, mais l'objectif de cet examen est de poser quelques jalons.

Question 1. On prend $\alpha = 1/3$, $\beta = 0.95$. Définissez les deux constantes C++ correspondantes, avec les noms et types qui conviennent.

Question 2. On prend $\{z_1, \ldots, z_n\} = \{0.9792, 0.9896, 1.0000, 1.0106, 1.0212\}$. Définissez un `vector` C++ (et non pas un tableau), appelé `z`, qui contient ces valeurs. Quelle librairie standard doit-on inclure pour utiliser des `vectors`?

Question 3. On définit encore les variables intermédiaires suivantes:

```
double capitalSteadyState     = std::pow(alpha*beta,1./(1-alpha));
double outputSteadyState      = std::pow(capitalSteadyState,alpha);
double consumptionSteadyState = outputSteadyState-capitalSteadyState;
```

Quelle est (mathématiquement) leur valeur? Quelle différence entre `1` et `1.` ?

Question 4. Dans le code de la Question 3, quels problèmes pourraient surgir? Donnez toutes les valeurs de α et β qui pourraient poser souci.

Question 5. Écrivez un bloc `if-then-else` qui capture les valeurs problématiques de α ou β, et affiche un message d'erreur le cas échéant.

Question 6. Si `alpha` et `beta` appartenaient à une classe C++, elles pourraient avoir une portée`private`, `protected`, ou `public`. Expliquez ces trois notions.

Question 7. On utilise pour résoudre le problème une "grille" de valeurs `gridCapital` de taille `17820`, telle que

```
gridCapital[i] = 0.5 * capitalSteadyState + 0.00001 * i;
```

Écrivez une boucle `for` qui remplit `gridCapital` comme ci-dessus. Quel type pourrait avoir `gridCapital`? Bonus : quel type pourrait avoir i?

Question 8. Écrivez une fonction `InitialiseGrid` qui prend en entrée `gridCapital`, et le remplit comme en Question 7. Utilisez les bons modificateurs (e.g. `const`, `&`, `&&`, `[]`, `static`, ...). Cette fonction est-elle pure ou impure, et pourquoi ?

Question 9. Nous allons utiliser des matrices, un type qui n'existe pas par défaut en C++. Mais on peut le créer ainsi:

```
template <std::size_t Rows, std::size_t Columns> using Matrix =
    std::array<std::array<double, Columns>, Rows>;
```

Nos matrices seront $N \times M$, avec $N = 17820$ et $M = 5$. Écrivez la signature d'une fonction `InitialiseExpectedValue` qui ne prend aucun argument et renvoie une `Matrix` de la bonne taille. Les détails de cette fonction sont sans intérêt ici (ils sont très simples).

Question 10. Écrivez une fonction const-safe, nommée `CopyMatrix`, qui prend en entrée une matrice A et retourne une nouvelle matrice qui est une copie de A. Quel est l'intérêt de la `const`-safety?

Question 11. Une fois fini, notre programme a la structure suivante:

1. Définition des constantes (Question 1)
2. Initialisation de la grille (Questions 7–8)
3. Répéter plusieurs fois:
   (a) Initialisation de la matrice d'espérances (Question 9)
   (b) Calcul de l'équation de Bellman V^i (non traité ici)
   (c) Copie de la nouvelle matrice (Question 10)
4. Le résultat final est la dernière matrice que l'on a calculé

Supposons le programme fini, nous devons maintenant le compiler. On utilise cette commande:

```
g++ -o rbc rbc.cpp -O3 --std=C++11
```

Expliquez tous les détails de cette commande.

Question 12. Nous avons placé tout notre code dans une fonction `OptimalBehaviour`, qui prend en entrée α et β, et retourne la séquence optimale $\{c_t, k_t\}$ pour tout $0 \leq t < T$. Nous voudrions maintenant utiliser cette fonction dans un programme annexe, plus gros. Que doit-on faire .

☐ Écrire une documentation détaillée qui précise les valeurs de α et β autorisées

☐ Mettre la fonction dans sa propre classe

☐ Mettre la fonction dans son propre namespace

☐ Créer un template avec des définitions de types

☐ Créer un fichier en-tête, et ajouter l'en-tête à la chaîne de compilation

☐ Créer un fichier en-tête, et ajouter le fichier de code à la chaîne de compilation

☐ Créer un fichier en-tête, et ajouter à la fois code et en-tête à la chaîne de compilation

Bonus: Écrivez une fonction récursive, pure, type-safe, `const`-safe, et documentée, qui calcule la racine carrée d'un nombre x à une précision donnée, utilisant que $r_k \xrightarrow{k\to\infty} \sqrt{x}$ où $r_{k+1} = \frac{1}{2}(r_k + x/r_k)$.