Numintro Dato: 23.10.2023

Afleveringsopgave

Ansvarlige: M. Bladt Aflevering: 27.10.2023 kl. 23:59

Tidsfrist og upload

Aflevering **senest** d. 27.10.2023 kl. 23.59.

Kode og output skal skrives ind i en PDF fil. Desuden skal koden uploades samlet som en .py fil KUN bestående af funktionerne (dvs. ikke jeres program, som jo også består af jeres kørsler etc., og ingen print statements inden i funktionerne). Formatet på funktionerne fremgår af delopgaverne, og er leveret også som Skabelon.py fil. I kan selv teste om filen virker ved at kalde den med source("Skabelon.py") kommandoen når I udregner jeres konkrete delopgaver.

Formatet er låst da vi tester programmerne automatisk/maskinelt. Navnet "Skabelon" SKAL erstattes af jeres studie ID (svenske nummerplader), så jeres fil med python koden kommer til at hedde e.g. abc123.py hvis abc123 er studie ID. Tilsvarende SKAL PDF filen navngives som abc123.pdf, igen hvis abc123 er studie ID. De to filer lægges i en mappe med navnet abc123, som derefter zippes og dermed uploades som abc123.zip.

Baggrund

Betragt et differentialligningssystem

$$\frac{d}{dt}\boldsymbol{A}(s,t) = \boldsymbol{A}(s,t)\boldsymbol{\Lambda}(t),$$

hvor $\boldsymbol{A}(s,t) = \boldsymbol{I}$ (identitetsmatricen) når $s = t$. Her er $\boldsymbol{A}(s,t)$ og $\Lambda(t)$ matricer af en givet orden p. Løsningen til denne ligning kendes ikke, men man har indført et symbol

$$\boldsymbol{A}(s,t) = \prod_s^t (\boldsymbol{I} + \Lambda(x)\mathrm{d}x)$$

som går under navnet produktintegralet af $\boldsymbol{\Lambda}(x)$. Navnet og notationen er foranlediget af approksimationen

$$\prod_s^t (\boldsymbol{I} + \Lambda(x)\mathrm{d}x) \approx \mathrm{e}^{\Lambda(s)h}\mathrm{e}^{\Lambda(s+h)h}\cdots\mathrm{e}^{\Lambda(t-2h)h}\mathrm{e}^{\Lambda(t-h)h} \tag{1}$$

for en inddeling af $[s,t]$ med skridtlængde h, hvor matrixeksponentieringen er defineret ved

$$\mathrm{e}^{\boldsymbol{A}x} = \sum_{n=0}^{\infty} \frac{x^n}{n!} \boldsymbol{A}^n$$

for en givet matrix $\boldsymbol{A}$. Hvis h er lille, så fås yderligere

$$\mathrm{e}^{\boldsymbol{A}h} \approx \boldsymbol{I} + \boldsymbol{A}h$$

op til en $O(h^2)$ fejl. Dette, sammenhold med (1), forklarer ideen bag notationen.

Hvis koefficientmatricen $\boldsymbol{\Lambda}(x) = \boldsymbol{\Lambda}$ er en konstant (matrix), så er løsningen til differentialligningssystemet

$$\frac{d}{dt}\boldsymbol{A}(s,t) = \boldsymbol{A}(s,t)\boldsymbol{\Lambda}, \quad \boldsymbol{A}(s,s) = \boldsymbol{I},$$

givet ved matrixeksponentieringen af Λ, i.e.

$$\boldsymbol{A}(s,t) = \exp\left(\boldsymbol{\Lambda}(t-s)\right) = \sum_{n=0}^{\infty} \frac{\boldsymbol{\Lambda}^n}{n!}(t-s)^n.$$

Opgave

Opgaven er praktisk motiveret af en forsikringsmodel, der anvendes praksis i de danske forsikringsselskaber. Denne fortolkning eller forståelse er dog hverken nødvendig eller anvendelig i besvarelsen af opgaven. Jeg fortsætter dog med at relatere de forskellige størrelser til de praktiske omstændigheder.

Vi betragter en matematisk model for en invalideforsikring. En person som tegner en forsikring antages at være rask når dette sker. Personen siges at være i "aktivtilstanden" eller tilstand 1. De andre tilstande som er vigtige for forsikringsselskabet er "invalidetilstand", givet ved tilstand 2, og "død", som tilstand 3.

Man beskriver nu dynamikken af en person gennem årene (tiden) v.h.a. en såkaldt Markov model med de tre tilstande. Det er en stokastisk model, hvor overgangssandsynlighederne $p_{ij}(s,t)$ angiver sandsynligheden for at en s–årig person som er i tilstand i vil være i tilstand j til tid (alder) $t > s$. Her kan i godt være lig med j. F.eks. er $p_{12}(40,60)$ sandsynligheden for at en 40–årig aktiv person vil være invalid som 60–årig. Både s og t er antaget at være reelle, så vi kan f.eks. også tale om en 45.746374–årig person.

Overgangssandsynlighederne arrangeres i en såkaldt overgangsmatrix

$$\boldsymbol{P}(s,t) = \begin{pmatrix} p_{11}(s,t) & p_{12}(s,t) & p_{13}(s,t) \\ p_{21}(s,t) & p_{22}(s,t) & p_{23}(s,t) \\ p_{31}(s,t) & p_{32}(s,t) & p_{33}(s,t) \end{pmatrix},$$

og det vides at denne er løsning til differentialligningssystemet (for fast s)

$$\frac{\partial}{\partial t}\boldsymbol{P}(s,t) = \boldsymbol{P}(s,t)\boldsymbol{\Lambda}(t), \quad \boldsymbol{P}(s,s) = \boldsymbol{I}, \tag{2}$$

hvor $\boldsymbol{I}$ er identitetsmatricen af dimension 3×3, og hvor $\boldsymbol{\Lambda}(t)$ er såkaldte intensitetsmatricer. D.v.s.,

$$\boldsymbol{P}(s,t) = \prod_{s}^{t}(\boldsymbol{I} + \boldsymbol{\Lambda}(x)\mathrm{d}x).$$

For mandlige forsikrede i Danmark bruges

$$\boldsymbol{\Lambda}(x) = \begin{cases} \begin{pmatrix} * & 0.0004 + 10^{4.54+0.06x-10} & 0.0005 + 10^{5.88+0.038x-10} \\ 2.0058 * \exp(-0.117x) & * & 2 \cdot (0.0005 + 10^{5.88+0.038x-10}) \\ 0 & 0 & 0 \end{pmatrix}, & x \leq 65, \\ \begin{pmatrix} * & 0.0004 + 10^{4.54+0.06x-10} & 0.0005 + 10^{5.88+0.038x-10} \\ 2.0058 * \exp(-0.117x) & * & 0.0005 + 10^{5.88+0.038x-10} \\ 0 & 0 & 0 \end{pmatrix}, & x > 65 \end{cases}$$

Diagonalelementerne indikeret med $*$ er givet ved minus summen af resten af elementerne i samme række, således at rækkesummerne alle bliver 0. F.eks. er den første $*$ for $x \leq 65$ givet ved

$$-\left(0.0004 + 10^{4.54+0.06x-10}\right) - \left(0.0005 + 10^{5.88+0.038x-10}\right).$$

Specifikke spørgsmål

A

Implementer ovenstående matrix-funktion $\boldsymbol{\Lambda}(x)$. Koden skal være på formen:

```
def Lambda(x):
#      kode til udregning af 3 x 3 matrix A=Lambda(x)
#      for et givet x som input
      return A
```

Som en test for funktionen burde den give (afrundet til 5 decimaler)

$$\boldsymbol{\Lambda}(50) = \begin{pmatrix} -0.01039 & 0.00387 & 0.00653 \\ 0.00578 & -0.01883 & 0.01305 \\ 0 & 0 & 0 \end{pmatrix}.$$

I jeres programmer bør I dog aldrig foretage afrundinger i mellemregningerne.

B

Implementer en 4. ordens Runge–Kutta metode til løsning af differentialligningssystemet (2) for $0 \leq s < t$. Denne skal have præcist følgende form

```
def Prodint(Lambda, s, t, n):
  # kode der udregner loesningen P = P(s,t)
    return P
```

hvor n angiver antal inddelinger af $[s,t]$, i.e. skridtlængde $h = (t-s)/n$.

Udregn

$$\boldsymbol{P}(40,70) = \texttt{prodint}(\texttt{Lambda},40,70,5000).$$

C

Find Richardson ekstrapolationen af **B**. baseret på N og $2N$, hvor $N = 1000$. Denne skal ikke implementeres, men der skal redegøres for beregningen i PDF filen.

Antag nu, at den forsikrede x–årige modtager betalinger med rate $r_k(x)$ hvis vedkommende er i tilstand k (negative betalinger kaldes for præmier). Over et interval på $[a,b]$ vil vedkommende derfor akkumulere

$$\int_a^b r_k(x)\mathrm{d}x$$

hvis tilstanden ikke ændres. Betragt så

$$\int_s^t p_{ik}(s,u)r_k(u)p_{kj}(u,t)\mathrm{d}u.$$

Dette er den forventede akkumulering af betalinger op til tid t fra tilstand k hvis den s–årige starter i tilstand i og slutter i tilstand j til tid t. Således er

$$\sum_k \int_s^t p_{ik}(s,u)r_k(u)p_{kj}(u,t)\mathrm{d}u$$

den samlede betaling til tid t den s–årige, der starter i tilstand i, kan forvente når hvis han er i tilstand j til tid t.

Hvis rentesatsen er på $r \geq 0$, så vil nutidværdien (i.e. til tid s) af den samlede forventede betaling være

$$\sum_k \int_s^t p_{ik}(s,u)e^{-r(u-s)}r_k(u)p_{kj}(u,t)\mathrm{d}u.$$

Den sidste formel er den (i,j)'te indgang i matricen

$$\boldsymbol{M}(s,t) = \int_s^t e^{-r(u-s)} \boldsymbol{P}(s,u) \boldsymbol{R}(u) \boldsymbol{P}(u,t) \mathrm{d}u,$$

hvor

$$\boldsymbol{R}(u) = \begin{pmatrix} r_1(u) & 0 & 0 \\ 0 & r_2(u) & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

D

Lad

$$\boldsymbol{A}(s,t) = \begin{pmatrix} \mathrm{e}^{-r(t-s)} \boldsymbol{P}(s,t) & \boldsymbol{M}(s,t) \\ \boldsymbol{0} & \boldsymbol{P}(s,t) \end{pmatrix}$$

være en 2×2 blok–matrix bestående af 3×3 matricer. Her er $\boldsymbol{0}$ en 3×3 matrix af nuller. Vis, at $\boldsymbol{A}(s,t)$ opfylder

$$\frac{\partial}{\partial t} \boldsymbol{A}(s,t) = \boldsymbol{A}(s,t) \begin{pmatrix} \boldsymbol{\Lambda}(t) - r\boldsymbol{I} & \boldsymbol{R}(t) \\ \boldsymbol{0} & \boldsymbol{\Lambda}(t) \end{pmatrix}, \quad \boldsymbol{A}(s,s) = \boldsymbol{I},$$

hvor $\boldsymbol{I}$ er identitetsmatricer af passende dimension. (Vink: brug kun Leibniz' regel og (2))

Betragt følgende forsikring. En 40–årig mand køber en forsikring, der ved invaliditet inden han fylder 65 sikrer ham 100,000 kr. årligt indtil pensionsalderen på 65. Både aktive og invalide overgår til pension på 100,000 kr. årligt efter de er fyldt 65. Lad $\mu = -r_1 = -r_1(x)$ angive den konstante præmierate (positiv) der skal betales af personen i tilstand 1 så længe $x \leq 65$. For tilstand 2 er $r_2 = r_2(x) = 1$ (målt i hundredetusinder) for alle aldre, mens $r_1 = r_x(x) = 1$ for $x > 65$. Dvs.

$$R(x) = \begin{cases} \begin{pmatrix} -\mu & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, & x \leq 65 \\ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, & x > 65 \end{cases}$$

Rentesatsen antages i det følgende at være $r = 0.01$.

E

Implementer $R(x)$ ovenfor. Den skal afhænge af μ, så den skal være på formen:

```
def R(x, mu):
  # kode til A=R(x) for et givet mu
    return A
```

Skriv dernæst en funktion som udregner $\boldsymbol{M}(s,t)$. Den skal være på formen

```
def M(s, t, Lambda, R, Prodint,mu, rint, n):
  # rint = r er renten
  # kode til udregning af RES=M(s,t)

    return RES
```

Udregn $M(40,70)$ med $\mu = 0.5$, $r = 0.01$ og $n = 5000$.

Ved reserven $V(s,t)$ forstås den forventede værdi af de akkumulerede betalinger til tid t uanset slutilstand. Så er

$$V(s,t) = \boldsymbol{e}_1' \boldsymbol{M}(s,t)\boldsymbol{e},$$

hvor $\boldsymbol{e}_1' = (1,0,0)$ og $\boldsymbol{e}' = (1,1,1)$. Postmultiplikationen med $\boldsymbol{e}$ summer ud over rækkerne, mens $\boldsymbol{e}_1'$ piller den første række ud. Dvs. $V(s,t)$ er den totale forventede betaling en aktiv s–årig akkumulerer indtil alder t.

F

Implementer en funktion som udregner $V(s,t)$. Koden skal være på formen:

```
def Reserve(t, TT, Lambda, R, prodint, mu, rint, n):
  # kode til udregning af tallet res=V(s,t)
  # prodint er jeres kodning af produktintegralet.
    return res
```

Udregn reserven hvis $\mu = 0.5$, $s = 40$, $t = 100$ og $r = 0.01$, i.e. 40–årig betaler årligt 50,000 kr. indtil pensionsalderen.

Ækvivalenspræmien er den værdi af μ for hvilken reserven bliver nul. Dette bliver fortolket som den "fair" præmie hvor hverken forsikringsselskab eller forsikrede vinder på arrangementet.

G

Find ækvivalenspræmien i hele kroner ved bisektionsmetoden for $s = 40$, $t = 100$ og $r = 0.01$. Proceduren skal implementeres på formen:

```
def Equiv_premium(a, b, age, agelimit, Lambda, R, Reserve, rint, n):
    # kode til at finde det mu som giver nul reserve
    return(mu)
```

Her er a og b initialpunkter for bisektionsmetoden.

Som alternativ til Runge–Kutta udregning af de involverede differentialligninger er ideen nu at approximere løsningerne med et produkt af matrixeksponentieringer i retning af (1). Rationalet bag denne ide er, at den anvendte matrixfunktioner $\lambda(x)$ er estimeret ud fra observerede intensiteter som typisk er konstanter over en periode på et år. Derfor kunne vi lige så godt have brugt de observerede intensiteter direkte og bruge (1). Dette vil vi nu gøre i det følgende i en række skridt.

H

Lad $\boldsymbol{A} = \{a_{ij}\}_{i,j=1,..,p}$ være en intensitetsmatrix og lad $\eta = \max_i(-a_{ii})$. Vis, at så er

$$\boldsymbol{P} = \boldsymbol{I} + \eta^{-1}\boldsymbol{A}$$

en stokastisk matrix, i.e. alle indgange er ikke negative og rækkerne summer til 1, og, at

$$\exp(\boldsymbol{A}x) = e^{-\eta x}\sum_{n=0}^{\infty}\frac{(\eta x)^n}{n!}\boldsymbol{P}^n. \tag{3}$$

I

Lad $\|\cdot\|$ være matriksnormen induceret af maximumsnormen. Vis, at

$$\|\exp(\boldsymbol{A}x) - e^{-\eta x}\sum_{n=0}^{\ell}\frac{(\eta x)^n}{n!}\boldsymbol{P}^n\| < \varepsilon \tag{4}$$

hvis

$$\sum_{n=0}^{\ell}\frac{(\eta x)^n}{n!}e^{-\eta x} > 1-\varepsilon.$$

J

Programmer en funktion i python til udregning af matrixeksponentiering af en matrix.

```
def Unif_matexp(A,x,eps):
    # input matrix A, skalar x og praecision eps
    #  output matrix M, hvis fejl er hoejst eps, maalt i matrixnormen
    # induceret af maximumsnormen
    return M
```

K

Skriv en python funktion som approksimerer produktintegralet, ved at approksimere dette som et produkt af exponentialfunktioner af konstante matricer. Mere præcist, vi inddeler et givet interval $[s,t]$ i $[s,s+1],[s+1,s+2],...,[s+k-1,s+k],[s+k,t]$, hvor k er det største heltal for hvilket $s+k<t$, og bruger

$$\prod_s^t (\boldsymbol{I}+\boldsymbol{\Lambda}(x)\mathrm{d}x)$$
$$\approx \exp(\boldsymbol{\Lambda}(s+0.5))\exp(\boldsymbol{\Lambda}(s+1.5))\cdots\exp(\boldsymbol{\Lambda}(s+k-0.5)\exp(\boldsymbol{\Lambda}(\tfrac{s+k+t}{2})(t-(s+k))).$$

Funktionen skal have følgende form

```
def Aprodint(Lambda,Unif_matexp,s,t,eps):
  # input kontinuert matrix funtion, interval s,t og praecision eps
  # output produktet af matrixeksponentieringer som approx til
  # produktintegralet
    return S
```

L

Genudregn reserven fra **F** hvor aprodint bruges i stedet for prodint i reservefunktionen.