GF7013 - Métodos Inversos Avanzados

Tarea 1

Muestreo de Variables Aleatorias

Fecha Publicación: Jueves 3 de abril de 2025
Fecha Última de Entrega: Viernes 25 de Abril de 2025

A lo largo del curso, mediante el desarrollo de las tareas y del mini proyecto final, se construirá un paquete de Python 3 llamado GF7013, el cual podrán seguir desarrollando para ser aplicado a muchos problemas de inversión que puedan presentarse en el futuro.

Se entrega una versión incipiente del paquete GF7013 la cual tiene una estructura similar a la siguiente:

```
GF7013
├── __init__.py
├── bin
│   ├── Tarea1
│   │   ├── Informe_Tarea1.ipynb
│   │   ├── P1.py
│   │   ├── P2.py
│   │   ├── P3.py
│   │   └── test_multinomial.py
│   ├── Tarea2
│   ├── Tarea3
│   └── Tarea4
├── model_parameters
├── models
├── probability_functions
│   ├── __init__.py
│   ├── likelihood
│   ├── pdf
│   │   ├── __init__.py
│   │   ├── pdf_base.py
│   │   ├── pdf_normal.py
│   │   └── pdf_uniform_nD.py
│   └── pmf
│       ├── __init__.py
│       ├── pmf_base.py
│       └── pmf_multinomial.py
└── sampling
    ├── __init__.py
    ├── catmip
    ├── metropolis
    ├── metropolis_in_parallel
    ├── rejection
    ├── resampling
    └── tmcmc
```

En GF7013, se encuentran 5 subcarpetas: `bin`, `model_parameters`, `models`, `probability_functions` y `sampling`. En `bin`, se desarrollarán los ejemplos que se ejecutarán en esta tarea y en las siguientes. La idea es poder colocar ejemplos que sean reproducibles por cualquier usuario y que permitan verificar el buen funcionamiento de los códigos programados. En `model_parameters` se diseñará una clase para representar los parámetros del modelo que se quiere estimar mediante un procedimiento inverso. En `models` se alojarán sub-paquetes que permiten definir diferentes modelos teóricos en los cuales se necesita realizar una inversión. En la carpeta `probability_functions` se incluirá código para generar muestras de diversas funciones de densidad de probabilidad (pdf: probability density functions), funciones de masa de probabilidad para variables discretas (pmf: probability mass functions), y otras funciones de probabilidad que se verán más adelante en el curso. En `sampling` irán los códigos que generan muestras de una variable aleatoria de manera indirecta , y en particular las técnicas de muestreo que se utilizarán para resolver problemas inversos.

En esta tarea se implementa clases de python que permiten evaluar funciones de probabilidad, verosimilitud, y generar muestras de variables aleatorias que siguen distribuciones de probabilidad clásicas. Las carpetas relevantes para esta tarea son `bin/Tarea1`, `probability_functions/pdf` y `probability_functions/pmf`. Para ello resuelva las preguntas a continuación, las que mejorarán la utilidad de los códigos proveídos y le permitirán comprender de mejor manera los algoritmos clásicos de generación de muestras de una variable aleatoria.

IMPORTANTE: El paquete GF7013 se irá aumentando de manera incremental a lo largo del curso. Por lo anterior es muy importante que no cambie la ubicación de los archivos ni carpetas dentro del paquete.

P1. Distribución Multinomial: un ejemplo de variables aleatorias discretas

En clases, se vio el ejemplo de una variable aleatoria discreta que pueden seguir una distribución multinomial, es decir que puede tomar Q estados discretos, cada uno con una cierta probabilidad. En el módulo `probability_functions/pmf/pmf_base.py` se entrega la definición de la clase base de una función de masa de probabilidad o *probability mass function (pmf)*. Asimismo, se entrega una versión funcional de la clase derivada de `pmf_base.py` que permite representar una variable aleatoria de la distribución multinomial. Para generar muestras de la distribución, se utiliza el algoritmo proveído por numpy.random (ver código y ayuda de la función `rng.multinomial`).

En esta pregunta se pide programar el método "análogo" (`method = 'analog'`) visto en clases para generar muestras de dicha variable aleatoria. Para ello:

P1.1. complete el código en el módulo `pmf_multinomial.py`.

P1.2. basándose en el ejemplo proveído en `bin/Tarea1/test_multinomial.py`, escriba un código en el módulo `bin/Tarea1/P1.py` que le permita comparar el conteo normalizado de las muestras que genera usando ambos métodos: el proveído "numpy", el programado por usted "analog" y la probabilidad de cada estado. Haga un gráfico de dicha comparación como el que se realiza en el ejemplo, pero con 3 barras por cada estado discreto de la variable aleatoria.

Variables aleatorias continuas

En el módulo `probability_functions/pdf/pdf_base.py` se entrega la clase de base de python para la representación de una función de dendisdad de probabilidad de una variable aleatoria continua (escalar o vectorial). Por favor estudie ese módulo y comprenda su contenido. Ponga atención a que se exigen funciones que entregan el valor de la verosimilitud (probabilidad no necesariamente normalizada) y el valor de la funcion de desidad de probabilidad, dado un valor específico de la variable aleatoria. Asimismo, se pide el logaritmo de dichos valores. Para esto último, NO tome el logaritmo de la cantidad anterior en la función de python, sino que debe trabajar las ecuaciones - tomando el logaritmo de la verosimilitud o probabilidad - para encontrar la expresión adecuada a programar en dichas funciones, de manera de no tener problemas numéricos debido a la finitud de la precisión de punto flotante del computador que usa para hacer los cálculos).

P2. Distribución Uniforme en N dimensiones

Para una variable aleatoria que sigue una distribución uniforme n-dimensional se pide:

P2.0 En su informe, indique la ecuación que define la fdp de la distribución uniforme en 1D. Luego considerando variables aleatorias independientes, indique cual es la fdp conjunta de la distribución uniforme en n dimensiones (i.e., para una variable aleatoria en $\mathbb{R}^n$). En ambos casos, indique claramente una expresión para la constante de normalización de la fdp.

P2.1 completar las funciones indicadas en el módulo `pdf_uniform_nD.py`. En particular, para la función `_draw` utilice el método inverso para generar muestras. Considere que el vector de variables aleatorias esta compuesto de variables aleatorias x_i que son independientes, cada una siguiendo una distribución uniforme U]a_i, b_i], donde los límites del intervalo pueden ser diferentes para cada variable aleatoria del vector. Escriba en su informe la ecuación de la función de probabilidad acumulada que ocupó.

P2.2 Use el módulo P2.py . Considere una variable aleatoria en $\mathbb{R}^2$, que sigue una distribución uniforme U(]-2,5] x]3.2,7]). Genere 1E5 muestras de dicha variable aleatoria y haga un histograma 2D (con mapa de colores mostrando las cuentas) de las muestras que aproxima la fdp conjunta, así como dos histogramas marginales (de cada una de las dos variables aleatorias) que aproximan las fdp marginales de cada variable.

P2.3 Repita el procedimiento del item anterior con 1E4 muestras y con 1E6 muestras. Comente sus resultados.

P3. Distribución Normal Multivariada

Para una variable aleatoria que sigue la distribución normal multivariada se pide:

P3.0 En su informe, indique la ecuación que define la fdp de la distribución normal N(μ, σ^2) en 1D. Luego, indique cual es la fdp conjunta de la distribución normal multivariada N($\boldsymbol{\mu}$, $\mathbf{C}$) (i.e., para una variable aleatoria en $\mathbb{R}^n$). En ambos casos, indique claramente una expresión para la constante de normalización de la fdp.

P3.1 complete las funciones indicadas en el módulo `pdf_normal.py`. En particular para la función `_draw` utilice el método relacional para generar realizaciones de una distribución normal N($\boldsymbol{\mu}$, $\mathbf{C}$), basado en el algoritmo proveído por numpy para generar números de una distribución normal estándar (canónica) N(0,1).

P3.2 Use el módulo P3.py . Considere una variable aleatoria en $\mathbb{R}^2$, que sigue una distribución normal multivariada N($\boldsymbol{\mu}$, $\mathbf{C}$) donde $\boldsymbol{\mu} = \begin{bmatrix} 0.5 & 3.0 \end{bmatrix}$ y $\mathbf{C} = \begin{bmatrix} 2 & 1 \\ 1 & 4 \end{bmatrix}$. Genere 1E5 muestras de dicha variable aleatoria y haga un histograma 2D (con mapa de colores mostrando las cuentas) de las muestras que aproxima la fdp conjunta, así como dos histogramas marginales (de cada una de las dos variables aleatorias) que aproximan las fdp marginales de cada variable. En los gráficos de los histogramas marginales (normalizados) añada una curva con los valores de la fdp marginal de cada variable. Calcule el promedio de las muestras y la matriz de covarianza de dichas muestras, compárelos con los parámetros $\boldsymbol{\mu}$ y $\mathbf{C}$ que definen la distribución normal multivariada.

P3.3 Repita el procedimiento del item anterior con 1E4 muestras y con 1E6 muestras. Comente sus resultados.

P3.4 Teniendo cuidado con la semilla en la generación de números aleatorios, ¿cómo diseñaría una función `test_draw()` para el módulo `pdf_normal.py` que devuelva el valor $True$ o $False$ si la generación de muestras de la normal multivariada funciona bien o mal? (tenga cuidado con la semilla del generador de números aleatorios y el número de muestras a considerar). Implemente dicha función y dentro del módulo P3.py y pruébela. Comente las dificultades encontradas y sus resultados.

Instrucciones

- La tarea se realizará en grupos de 4 (POR FAVOR ENVIAR DEFINICIÓN DE GRUPOS POR EMAIL AL PROFESOR)
- Se prohíbe la colaboración entre grupos.
- La nota de la Tarea para cada integrante del grupo se calculará de la siguiente manera

$$Nota = NC * F_i \tag{1}$$

donde NC es la nota de la corrección de la Tarea (hecha por el profesor) y F_i es un factor calculado en función del promedio de las notas con que cada integrante del grupo evalúa a los otros integrantes del grupo (se hará a través de ucursos y es confidencial). La idea de F_i es proveer una instancia de evaluación de la participación y compromiso con el desarrollo de la tarea entre los integrantes de cada grupo. F se calcula según la siguiente fórmula

$$F_i = \text{máx}\Big(0.5,\ \frac{NG_i}{NG_{\text{prom}}}\Big)$$

donde NG_i es la nota del integrante del grupo que resulta de promediar las notas asignadas por sus compañeros en la co-evaluación confidencial. y NG_{prom} es el promedio de los NG_i del grupo.

- Todo el material necesario para realizar la tarea se encuentra en una carpeta llamada `Tarea1_GRUPO_XX`, en sus apuntes de clases y en las referencias bibliográficas del programa del curso..
- Copiar la estructura de carpetas en una carpeta en donde el nombre/numero del grupo reemplaza XX.

- Al entregar la tarea deben entregar, el informe debe ser escrito dentro de los notebook de python que se les entrega (`Informe_Tarea1.ipynb`), esto de manera ordenada y se entienda claramente los gráficos y las respuestas a cada parte de la tarea. De manera alternativa, si lo desea puede escribir un informe en Latex, Word o el software de su elección, siguiendo los mismos lineamientos. A u-cursos se debe subir un archivo comprimido que tenga la estructura de carpetas `Tarea1_GRUPO_XX` con los códigos y notebooks desarrollados al interior (y el informe en formato PDF si no respondió directamente en los notebooks).
- No es necesario una introducción teórica en el informe, pero deben mostrar el desarrollo cuando se pida encontrar expresiones matemáticas o formulaciones particulares de los problemas (escribir las ecuaciones correspondientes y como las transforman cuando aplique). Si una celda del notebook esta en modo "MarkDown", en ella se puede escribir comandos de latex encerrados entre dobles signos pesos. Por ejemplo,

  ```
  $$ \frac{1}{2} \mathbf{G} $$
  ```

  (los espacios son importantes) queda $\frac{1}{2}\mathbf{G}$.
- En los horarios de clase auxiliar se dedicará tiempo a responder preguntas que se tengan acerca del desarrollo de la tarea. De la misma manera estaré disponible para responder consultas presenciales, por correo electrónico o Zoom de ser necesario.
- Asimismo es una excelente práctica el escribir códigos que sean modulares de manera de poder reutilizar la mayor cantidad de veces ese código (obviamente sin abusar).
- Para hacer histogramas 2D ver https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.hist2d.html
- **Se aceptan entregas atrasadas pero por cada día hábil de atraso se resta 0.5 pts a la nota final de la tarea. (día habil = {lunes, martes, miércoles, jueves, viernes} que no sea feriado)**

Exito!!!..