MAT-468: Sesión 9, Elementos de Simulación II

Felipe Osorio

http://fosorios.mat.utfsm.cl

Departamento de Matemática, UTFSM

Método de composición

Este método asume que F puede ser escrita como una mezcla de funciones de distribución $\{G_i\}$, esto es

$$F(x) = \sum_{i=1}^{M} p_i \, G_i(x),$$

donde $p_i > 0$ y $\sum_{i=1}^{M} p_i = 1$.

Sea $X_i \sim G_i$ y sea Y una v.a. discreta con

$$P(Y = i) = p_i,$$

independiente de X_i, $i = 1, \ldots, M$. Entonces una v.a.X con distribución F puede ser representada como

$$X = \sum_{i=1}^{M} X_i \, I_{\{Y=i\}}.$$

Método de composición

Para generar $X \sim F$ el método de composición utiliza el siguiente algoritmo:

(a) Generar una v.a. discreta Y como

$$P(Y = i) = p_i, \qquad i = 1, \ldots, M.$$

(b) Dado $Y = i$, generar $X \sim G_i$.

Observación:

Esta representación permite definir algoritmos bastante eficientes para la generación de variables aleatorias.[1]

[1] Aunque también es caso específico

Método de composición: t de Student

Ejemplo: Generación de v.a. t de Student

Varios tipos de distribuciones pueden ser escritos como

$$f(x) = \int g(x,y)\,\mathrm{d}y = \int h_1(x|y)h_2(y)\,\mathrm{d}y,$$

con h_1 y h_2 las densidades condicional y marginal de $X|Y=y$ e Y, respectivamente.

Por ejemplo, para la distribución t de Student, tenemos

$$X|Y=y \sim \mathsf{N}(\mu, \sigma^2/y), \qquad Y \sim \mathsf{Gama}(\nu/2, \nu/2),$$

es decir, $X \sim t(\mu, \sigma^2, \nu)$, $\nu > 0$.

Ejemplo: Generación de v.a. $\chi^2_r(\lambda)$

La función de densidad de una distribución $\chi^2_r(\lambda)$ puede ser escrita como

$$f(q; r, \lambda) = \sum_{j=0}^{\infty} \Big(\frac{e^{-\lambda}\lambda^j}{j!}\Big) g(q; r + 2j, 0),$$

donde $g(q; s, 0)$ representa la función de densidad de una v.a $\chi^2_s(0)$.

Observación: Esta representación no es muy útil para definir un RNG. Por otro lado, el método estándar para generar v.a. desde $\chi^2_r(\lambda)$ es basado en la propiedad

$$Z \sim \chi^2_{r-1}(0), \qquad Y \sim \mathsf{N}(\sqrt{\lambda}, 1),$$

y hacemos la transformación

$$Z + Y^2 \sim \chi^2_r(\lambda).$$

Método de aceptación-rechazo

- Los métodos de inversión y composición son directos, pues utilizan directamente F (distribución objetivo).
- En algunos contextos es imposible explotar las propiedades de F para derivar un RNG.
- Esta clase de métodos requiere conocer la forma de la densidad f de interés (salvo una constante multiplicativa).
- La clave del método es usar una densidad más simple g para simular (densidad instrumental).

El método de aceptación-rechazo se basa en la siguiente idea, note que

$$f(x) = \int_0^{f(x)} \mathrm{d}u.$$

De este modo, f corresponde a la densidad marginal desde la conjunta

$$(X, U) \sim \mathsf{U}\{(x, u) : 0 < u < f(x)\}.$$

Resultado 4

Simular $X \sim f(x)$ es equivalente a simular[2]

$$(X, U) \sim \mathsf{U}\{(x, u) : 0 < u < f(x)\}.$$

[2] Esto puede no ser sencillo

Idea:
Se puede simplificar la simulación del par (X, U) a un conjunto mayor donde la simulación sea sencilla y luego tomar un par tal que la restricción sea satisfecha.

Considere g una función de densidad arbitrarea tal que,

$$\phi(x) = Mg(x)$$

mayoriza $f(x)$ para algún M, esto es, $\phi(x) \geq f(x)$ para todo x. Note que $M \geq 1$.

Resultado 5

Sea $X \sim f(x)$ y considere $g(x)$ una función de densidad tal que $f(x) \leq Mg(x)$ para alguna constante $M \geq 1$. Entonces para simular $X \sim f$ es suficiente hacer

$$Y \sim g, \qquad U|Y = y \sim \mathsf{U}(0, Mg(y)),$$

hasta que $0 < u < f(y)$.

Método de aceptación-rechazo

Algoritmo: Aceptación-Rechazo

1. Generar $x \sim g$.
2. Generar $u \sim \mathrm{U}(0, 1)$ independiente de x.
3. Aceptar $y = x$ si $u \leq f(x)/Mg(x)$.
4. En otro caso volver a 1.

EX UMBRA IN SOLEM

Previo: Generando desde la distribución Laplace

Considere $X \sim \text{Laplace}(\mu, \phi)$, en cuyo caso tenemos

$$f(x; \mu, \phi) = \frac{1}{2\phi} \exp\Big(- \frac{-|x-\mu|}{\phi}\Big),$$

y es fácil notar que

$$\begin{aligned} F(x) &= \begin{cases} \frac{1}{2}\exp(\frac{x-\mu}{\phi}), & x < \mu, \\ 1 - \frac{1}{2}\exp(-\frac{x-\mu}{\phi}), & x \geq \mu, \end{cases} \\ &= \frac{1}{2}\Big\{1 + \operatorname{signo}(x-\mu)\Big(1 - \exp\Big(-\frac{|x-\mu|}{\phi}\Big)\Big)\Big\}. \end{aligned}$$

Así,

$$F^{-1}(u) = \mu - \phi \operatorname{signo}(u - \tfrac{1}{2}) \log(1 - 2|u - \tfrac{1}{2}|).$$

Finalmente podemos generar observaciones desde $X \sim \text{Laplace}(\mu, \phi)$ considerando:

$$x = \mu - \phi \operatorname{signo}(u) \log(1 - 2|u|),$$

para $u \sim \text{U}(-\frac{1}{2}, \frac{1}{2})$.

Note que

$$\frac{1}{2}(|x|-1)^2 = \frac{x^2}{2} - |x| + \frac{1}{2} \geq 0 \quad \Rightarrow \quad \frac{x^2}{2} \geq |x| - \frac{1}{2}$$

es decir

$$\exp(-x^2/2) \leq \exp(-|x| + 1/2) \quad \Rightarrow \quad \frac{1}{\sqrt{2\pi}} e^{-x^2/2} \leq \frac{1}{\sqrt{2\pi}} e^{-|x|+1/2} \qquad (1)$$

Ahora,

$$\sqrt{\frac{e}{2\pi}} e^{-|x|} = \sqrt{\frac{2e}{\pi}} \, \frac{1}{2} e^{-|x|} = M g(x),$$

donde $M = \sqrt{2e/\pi} \approx 1.3155$.

Así desde (1), tenemos que para $X \sim \mathsf{N}(0,1)$,

$$f(x) \leq M g(x),$$

el algoritmo de aceptación-rechazo, requiere de ≈ 1.3 v.a. uniformes para generar una v.a. normal.

Algoritmo: RNG $\mathsf{N}(0,1)$ desde Laplace$(0,1)$

1. Repetir:
   a. Generar $X \sim \text{Exp}(1)$ y U, V v.a. $\mathsf{U}(0,1)$ independientes.
   b. Si $U < \frac{1}{2}$ hacer $X \leftarrow -X$ (es decir, $X \sim$ Laplace$(0,1)$)

   hasta que $V \frac{1}{\sqrt{2\pi}} \exp(\frac{1}{2} - |x|) \leq \frac{1}{\sqrt{2\pi}} \exp(-x^2/2)$.
2. Retornar X.

Observaciones:

- La constante $(2\pi)^{-1/2}$ cancela a ámbos lados en la etapa de aceptación, además es conveniente tomar logaritmos.
- En el paso 1.b. no hay necesidad de hacer un cambio de signo para una v.a. que será rechazada (podemos evitar simular la v.a. U).

Generando $\mathsf{N}(0,1)$ usando la densidad Laplace

Algoritmo: RNG $\mathsf{N}(0,1)$ desde Laplace$(0,1)$

1. Repetir:

   a. Generar $X \sim \mathsf{Exp}(1)$ y $V \sim \mathsf{U}(-1,1)$ independientes.

   hasta que $(X-1)^2 \leq -2\log(|V|)$.

2. Retornar $X \leftarrow \mathrm{signo}(V)X$.

Elección de M óptimo

Para f y g dadas, la constante M debe escogerse como:[3]

$$M = \sup_x \frac{f(x)}{g(x)}.$$

Usualmente esto requiere resolver un problema de optimización. El M óptimo es aquél más cercano a 1.

Ejemplo: RNG $\mathbf{N(0,1)}$ desde Cauchy$(\mathbf{0}, \boldsymbol{\phi})$

Considere

$$g_\phi(x) = \frac{\phi}{\pi} \frac{1}{x^2 + \phi^2}.$$

En este caso (tarea),

$$M_\phi = \begin{cases} \frac{\sqrt{2\pi}}{\phi e} \exp(\phi^2/2), & \phi < \sqrt{2}, \\ \phi\sqrt{\pi/2}, & \phi > \sqrt{2}. \end{cases}$$

[3] Es decir f y g deben tener el mismo soporte.

Generando $\mathsf{N}(0,1)$ usando la densidad Cauchy

Algoritmo: RNG $\mathsf{N}(0,1)$ desde Cauchy$(0,\phi)$

0. Hacer $\alpha \leftarrow \sqrt{e}/2$.
1. Repetir:
   a. Generar $U, V \sim \mathsf{U}(0,1)$ independientes.
   b. Hacer $X \leftarrow \tan(\pi V)$, $S \leftarrow X^2$ (es decir, $X \sim$ Cauchy$(0,1)$)

   hasta que $U \leq \alpha(1+S)e^{-S/2}$.
2. Retornar X.

Observación:

- Aunque la constante de aceptación (M) es cercana a 1.4, este algoritmo no es recomendable.
- Tanto la generación de v.a. Cauchy como el paso de aceptación son caros.

Método de razón de uniformes

Este procedimiento corresponde a un caso particular del método de aceptación-rechazo, donde se realiza un mejor control de la cola.

Resultado 1 (Kinderman y Monahan, 1977)

Si el punto (U,V) es uniformemente distribuído sobre la región

$$C_h = \{(u,v) : 0 < u < h^{1/2}(v/u)\}.$$

Entonces la razón $X = V/U$ tiene una densidad proporcional a $h(x)$.

Método de razón de uniformes

- La idea del método es generar uniformemente sobre la región C_h.
- Existe alguna ventaja si C_h se ajusta a una región fácil de simular.
- Usualmente se trabaja con la siguiente forma para $(u(x), v(x))$,

$$u(x) = \sqrt{h(x)}, \qquad v(x) = x\sqrt{h(x)},$$

y se considera el rectángulo:

$$\{(u, v) : 0 \leq u \leq b, c \leq v \leq d\},$$

donde

$$b = \max_x \sqrt{h(x)}, \quad c = \min_x x\sqrt{h(x)}, \quad d = \max_x x\sqrt{h(x)}.$$

Algoritmo: Razón de uniformes

1. Repetir:

    a. Generar $U, V \sim \mathsf{U}(0,1)$ independientes.

    b. Hacer

    $$\begin{aligned} U_1 &= b\,U, \\ V_1 &= c + (d - c)V, \end{aligned}$$

    (o sea, $U_1 \sim \mathsf{U}(0,b)$, $V_1 \sim \mathsf{U}(c,d)$).

    c. Hacer $X = V_1/U_1$.

    hasta que $U_1^2 \leq h(X)$.

2. Retornar X.

Método de razón de uniformes: RNG $\mathsf{N}(0,1)$

Considere simular desde $\mathsf{N}(0,1)$, en cuyo caso $h(x) = \exp(-x^2/2)$, y

$$b = \sup_x \sqrt{h(x)} = 1, \quad c = \inf_x x\sqrt{h(x)} = -\sqrt{2/e}, \quad d = \sup_x x\sqrt{h(x)} = \sqrt{2/e},$$

de este modo, obtenemos el algoritmo:

Algoritmo: RNG $\mathsf{N}(0,1)$ mediante ROU

1. Repetir:
   a. Generar $U, V \sim \mathsf{U}(0,1)$ independientes.
   b. Hacer $W = (2V-1)\sqrt{2/e}$, $X = W/U$.

   hasta que $-4U^2\log(U) \geq W^2$.
2. Retornar X.

Advertencia: Distribución logística y Laplace

Para la distribución logística, tenemos

$$F(x) = \frac{1}{1 + e^{-x}},$$

así podemos generar $X \sim \text{Logistica}(0, 1)$ usando

$$F^{-1}(u) = \log(u/(1 - u)).$$

Sin embargo, el siguiente código en R

```
x <- log(runif(1) / (1 - runif(1))
```

puede tener consecuencias impredecibles.[4]

[4] Este código simula desde la distribución Laplace$(0, 1)$

Sabemos que $\boldsymbol{X} \sim \mathsf{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ tiene densidad

$$f(\boldsymbol{x}) = |2\pi\boldsymbol{\Sigma}|^{-1/2} \exp\big\{ -\tfrac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\big\}.$$

Además, si $\boldsymbol{Z} \sim \mathsf{N}_p(\boldsymbol{0}, \boldsymbol{I})$, entonces

$$\boldsymbol{B}\boldsymbol{Z} + \boldsymbol{\mu} \sim \mathsf{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \qquad \boldsymbol{\Sigma} = \boldsymbol{B}\boldsymbol{B}^\top.$$

Así las distintas variantes de este algoritmo dependen del tipo de factorización usado para obtener $\boldsymbol{\Sigma} = \boldsymbol{B}\boldsymbol{B}^\top$. Por ejemplo, la factorización Cholesky, descomposición espectral o SVD.

Generador desde $\mathsf{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$

Algoritmo: RNG $\mathsf{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$

1. Generar p observaciones independientes desde $\mathsf{N}(0, 1)$, digamos $Z_1, \dots, Z_p$
2. Calcular una factorización $\boldsymbol{\Sigma} = \boldsymbol{G}^\top \boldsymbol{G}$.
3. Aplicar la transformación $\boldsymbol{X} = \boldsymbol{G}^\top \boldsymbol{Z} + \boldsymbol{\mu}$.

Usualmente no se desea un único vector aleartorio, sino una matriz $\boldsymbol{X} \in \mathbb{R}^{n \times p}$. En cuyo caso podemos modificar los pasos 1 y 3, como[5]

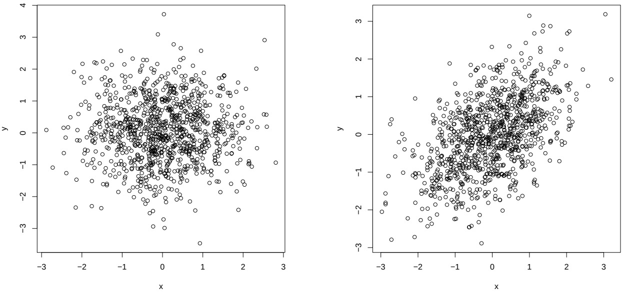
Algoritmo′: RNG $\mathsf{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$

1′. Generar una matriz $n \times p$ con np observaciones desde $\mathsf{N}(0, 1)$.

3′. Hacer $\boldsymbol{X} = \boldsymbol{Z}\boldsymbol{G} + \boldsymbol{1}_n \boldsymbol{\mu}^\top$, con $\boldsymbol{1}_n = (1, \dots, 1)^\top$.

[5] ver funciones: `mvrnorm` (MASS), `rmvnorm` (mvtnorm), `rmnorm` (heavy)

Datos generados desde $\mathbf{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$



Código en C: rutina rmnorm (heavy)

```c
void
rand_norm(double *y, int *pdims, double *center, double *Scatter)
{   /* multivariate normal random generation */
    DIMS dm;
    char *side = "L", *uplo = "U", *trans = "T", *diag = "N";
    double one = 1.;
    int i, inc = 1, info = 0, job = 1;

    dm = dims(pdims);
    GetRNGstate();

    chol_decomp(Scatter, dm->p, dm->p, job, &info);
    if (info)
       error("DPOTRF in cholesky decomposition gave code %d", info);

    rand_spherical_norm(y, dm->n, dm->p);

    F77_CALL(dtrmm)(side, uplo, trans, diag, &(dm->p), &(dm->n), &one, Scatter,
                    &(dm->p), y, &(dm->p));

    for (i = 0; i < dm->n; i++) {
        F77_CALL(daxpy)(&(dm->p), &one, center, &inc, y, &inc);
        y += dm->p;
    }

    PutRNGstate();
    dims_free(dm);
}
```

RNG para la distribución uniforme sobre la esfera

La esfera p-dimensional es el conjunto de todos los puntos $\boldsymbol{x} \in \mathbb{R}^p$, tal que

$$\|\boldsymbol{x}\| = (\boldsymbol{x}^\top \boldsymbol{x})^{1/2} = 1.$$

Un método para generar $\boldsymbol{U} \sim \mathcal{S}^{p-1}$ usa la propiedad:

Propiedad 1

Si $Z_1, \dots, Z_p \overset{\text{ind}}{\sim} \mathsf{N}(0,1)$, entonces $\boldsymbol{U} = (U_1, \dots, U_p)^\top$, definido como:

$$\boldsymbol{U} = \frac{\boldsymbol{Z}}{\|\boldsymbol{Z}\|}, \qquad \text{es decir} \qquad U_j = \frac{Z_j}{\{Z_1^2 + \cdots + Z_p^2\}^{1/2}}, \quad j = 1, \dots, p,$$

sigue una distribución uniforme sobre la esfera unitaria.

Generador desde $\mathcal{S}^{p-1}$

Algoritmo: RNG $\mathcal{S}^{p-1}$

1. Generar $Z_1, \dots, Z_p \overset{\text{ind}}{\sim} \mathsf{N}(0, 1)$.
2. Calcular $\|\boldsymbol{Z}\| = (Z_1^2 + \cdots + Z_p^2)^{1/2}$.
3. Hacer $U_i = Z_i / \|\boldsymbol{Z}\|$, para $i = 1, \dots, p$.
4. Retornar $\boldsymbol{U} = (U_1, \dots, U_p)^\top$.

Observación:

Este algoritmo permite generar vectores aleatorios $\boldsymbol{X} \sim \mathsf{EC}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma}; g)$[6] usando que

$$\boldsymbol{X} \overset{\mathsf{d}}{=} \boldsymbol{\mu} + R\boldsymbol{B}\boldsymbol{U},$$

donde la v.a $R \geq 0$ es independiente de $\boldsymbol{U} \sim \mathcal{S}^{p-1}$ y $\boldsymbol{\Sigma} = \boldsymbol{B}\boldsymbol{B}^\top$.

[6] con densidad $f(\boldsymbol{x}) = |\boldsymbol{\Sigma}|^{-1/2} g((\boldsymbol{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}))$

Código en C: rutina `rsphere` (heavy)

```c
void
rand_unif_sphere(double *y, int n, int p)
{   /* random vector generation uniformly on the sphere */
    int i, j, inc = 1;
    double radial;

    for (i = 0; i < n; i++) {
        for (j = 0; j < p; j++)
            y[j] = norm_rand();
        radial = 1. / F77_CALL(dnrm2)(&p, y, &inc);
        F77_CALL(dscal)(&p, &radial, y, &inc);
        y += p;
    }
}
```

Datos generados desde $\mathcal{S}^{p-1}$