

Métodos de Monte Carlo Sequenciais (SMC)

Trabalho - EST5514 - Simulação Estocástica

Alunos: Cleiton Moya de Almeida Kluyvert Monteiro Souza

Prof.^a: Daiane A. Zuanetti

Programa Interinstitucional de Pós-Graduação em Estatística UFSCar-USP
(PIPGEs)

28 de novembro de 2024

Roteiro

Introdução

Importance Sampling (IS)

Sequential Importance Sampling (SIS)

Sequential Monte Carlo (SMC)

Conclusões

Introdução

Introdução

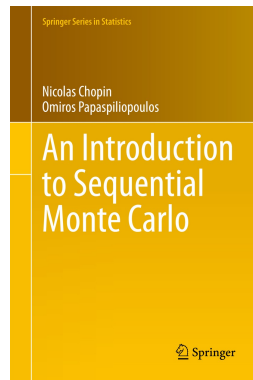
- ▶ **Sequential Monte Carlo (SMC):** Conjunto de métodos para inferência estatística aproximada, predominantemente bayesiana. (Naesseth et al., 2019).
 - Métodos baseados em *Importance Sampling* e reamostragem.
- ▶ **Primeiros trabalhos:** métodos para inferência *online* em modelos não lineares e/ou não-gaussianos em espaço de estados (SSM);
 - **Sequential Importance Sampling:** (Handschin & Mayne, 1969)
 - **Sequential Monte Carlo:** (Gordon et al., 1993)
 - Também denominado **filtro de partículas**.

Aplicações atuais

- ▶ Apesar de inicialmente desenvolvidos no contexto de SSM, e almejando inferência *online*, atualmente os métodos SMC são utilizados em diversas aplicações, inclusive para inferência *offline*.
- ▶ Naesseth et al. (2019) mencionam:
 - Programação probabilística;
 - Modelos gráficos probabilísticos;
 - Inferência variacional;
 - Avaliação de inferências;
 - Inferência bayesiana não-paramétrica.

Aplicações atuais

- ▶ Em (Chopin & Papaspiliopoulos, 2020, Cap. 3):
 - Simulação de eventos raros;
 - Simulação por têmpera;
 - Útil para distribuições multi-modais;
 - Otimização não-convexa (via têmpera);
 - *Likelihood-free inference*, algoritmos ABC.
- ▶ Este livro é uma introdução avançada:
 - SMC é visto como uma aproximação Monte Carlo para modelos Feynman-Kac.
- ▶ Área de pesquisa bastante ativa e densa;
 - **Neste trabalho:** versões básicas dos métodos e aplicações simples.



Importance Sampling (IS)

IS normalizado

- ▶ **Ideia:** distribuição alvo é aproximada de forma ponderada através de uma distribuição proposta e pesos (*importance weights*);.
- ▶ **Formulação do problema** (Zhou, 2022):
 - Seja $X \sim f(x)$, $x \in D$, onde $f(\cdot)$ é a f.d.p. (normalizada) de uma distribuição alvo na qual não sabemos gerar amostras.
 - Queremos calcular o valor esperado de uma determinada função $h(\cdot)$:

$$\mathbb{E}_f [h(X)] = \int_D h(x)f(x)dx \quad (1)$$

- Seja $g(x)$ a f.d.p (normalizada) de uma distribuição proposta com suporte S tal que $D \subset S$.

IS normalizado

► Podemos escrever:

$$\begin{aligned}\mathbb{E}_f[h(X)] &= \int_D h(x)f(x)dx \\ &= \int_S h(x)\frac{f(x)}{g(x)}g(x)dx \\ &= \mathbb{E}_g\left[h(X)\frac{f(X)}{g(X)}\right] \\ &\approx \frac{1}{N}\sum_{n=1}^N h\left(x^{(n)}\right)\underbrace{\frac{f\left(x^{(n)}\right)}{g\left(x^{(n)}\right)}}_{w\left(x^{(n)}\right):=w^{(n)}}, \quad x^{(n)} \stackrel{iid}{\sim} g.\end{aligned}\tag{2}$$

Algoritmo

Algorithm 1 Importance Sampling normalizado

- 1: Amostre $x^{(n)} \stackrel{iid}{\sim} g(\cdot)$ para $n = 1 \dots, N$
 - 2: Calcule os pesos correspondentes: $w^{(n)} = \frac{f(x^{(n)})}{g(x^{(n)})}$, para $n = 1, \dots, N$
 - 3: **return** (\mathbf{x}, \mathbf{w})
-

IS sem constantes de normalização

- ▶ No algoritmo anterior, assumimos que $f(x)$ e $g(x)$ são funções (densidades) normalizadas, ou seja, integram 1;
- ▶ Sejam $\tilde{f}(x)$ e $\tilde{g}(x)$ versões não-normalizadas de $f(x)$ e $g(x)$, respectivamente, com constantes de normalização Z_f e Z_g desconhecidas:
 - $f(x) = \tilde{f}(x)/Z_f$ e $g(x) = \tilde{g}(x)/Z_g$;
- ▶ Usando o mesmo raciocínio anterior, podemos modificar o algoritmo para utilizamos as versões não-normalizadas de f , g ou ambas;
- ▶ Esta versão é por vezes chamada de *auto-normalized importance sampling* (Chopin & Papaspiliopoulos, 2020).

Algoritmo

Algorithm 2 Importance Sampling auto-normalizado

- 1: Amostre $x^{(n)} \stackrel{iid}{\sim} \tilde{g}(\cdot)$, para $n = 1, \dots, N$
 - 2: Calcule os pesos não-normalizados: $\tilde{w}^{(n)} = \frac{\tilde{f}(x^{(n)})}{\tilde{g}(x^{(n)})}$, para $n = 1, \dots, N$
 - 3: Normalize os pesos: $w^{(n)} = \frac{\tilde{w}^{(n)}}{\sum_{n=1}^N \tilde{w}^{(n)}}$, para $n = 1, \dots, N$
 - 4: **return** $(\mathbf{x}, \tilde{\mathbf{w}}, \mathbf{w})$
-

► Neste caso,

$$\mathbb{E}_f[h(X)] \cong \sum_{n=1}^N h(x^{(n)}) w^{(n)} \quad (3)$$

Estimativa da constante de normalização por IS

- ▶ Suponha que queremos encontrar a constante de normalização Z_f de alguma função não normalizada $\tilde{f}(x)$.
- ▶ Se utilizarmos como proposta uma função densidade (normalizada) $g(x)$, podemos estimar Z_f por IS:

$$\hat{Z}_f = \frac{1}{N} \sum_{n=1}^N w^{(n)} \quad (4)$$

- ▶ Prova:

$$Z_f = \int \tilde{f}(x) dx = \int \frac{\tilde{f}(x)}{g(x)} g(x) dx = \int \tilde{w}(x) g(x) dx = \mathbb{E}_g[\tilde{w}(X)] \approx \frac{1}{N} \sum_{n=1}^N \tilde{w}^{(n)}$$

Geração de amostras - IS com reamostragem

- ▶ Dada a amostra $\{x^{(1)}, \dots, x^{(N)}\}$ obtidas de $g(x)$;
- ▶ E os pesos correspondentes $\{w^{(1)}, \dots, w^{(N)}\}$;
- ▶ Se reamostrarmos $x^{(*n)}$ de $\{x^{(1)}, \dots, x^{(N)}\}$ com reposição e com probabilidade igual ao peso normalizado correspondente, isto é,

$$\mathbb{P} \left[x^{(*n)} = x^{(k)} \mid \{x^{(1)}, \dots, x^{(N)}\} \right] = w^{(k)} \quad (5)$$

- ▶ Então, a distribuição de $\{x^{(*1)}, \dots, x^{(*N)}\}$ é aproximadamente a distribuição alvo f quando N é grande.

Exemplo

- ▶ Seja $f(x)$ a f.d.p. da distribuição Normal Absoluta, dada por

$$f(x) = \sqrt{\frac{2}{\pi}} e^{-\frac{x^2}{2}} \quad (6)$$

- $X = |Z|$, com $Z \sim \mathcal{N}(0, 1)$.
- ▶ Suponha que não conhecemos a constante de normalização $Z_f = \sqrt{\frac{\pi}{2}}$;
- ▶ Também, não sabemos calcular analiticamente $\mathbb{E}_f[X]$;
- ▶ Queremos estimar, por IS:
 - $\mathbb{E}_f[X] = \mu_f$;
 - Z_f ;
 - Amostras da distribuição f .

Exemplo

- ▶ Distribuição alvo (função não normalizada): $\tilde{f}(x) = e^{-\frac{x^2}{2}}$
- ▶ Distribuição proposta (normalizada): $g(x) = 2e^{-2x}$ ($\text{Exp}(\lambda = 2)$);
- ▶ Pesos não-normalizados:
$$\tilde{w}(x) = \frac{\tilde{f}(x)}{g(x)} = \frac{1}{2} \exp\left(-\frac{x^2}{2} + 2x\right)$$

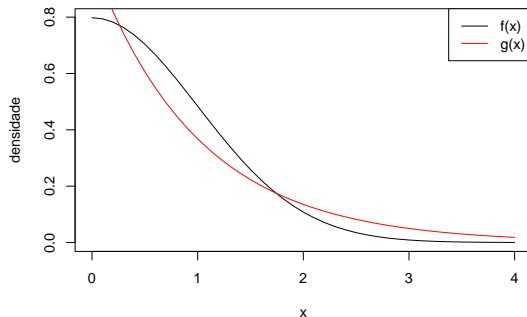


Figura 1: Funções alvo $f(x)$ e proposta $g(x)$

Exemplo

- ▶ $Z_f = \sqrt{\frac{\pi}{2}} \approx 1.253314$
- ▶ Com a função `integrate` do R:
 $\hat{\mu}_f = 0.7978846$
- ▶ Com IS:

N	$\hat{\mu}_f$	\hat{Z}_f
1.000	0.8451311	1.306103
10.000	0.8062410	1.261553
100.000	0.7972778	1.251665

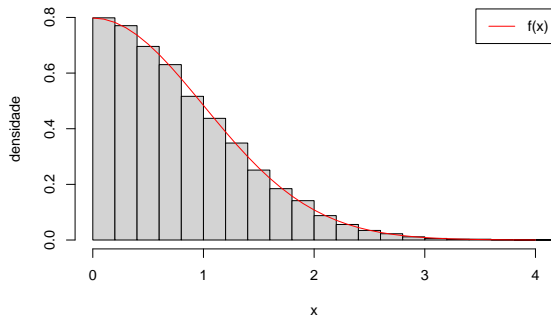


Figura 2: Hist. das amostras geradas (reamostragem) e função densidade de $f(x)$

Sequential Importance Sampling (SIS)

Sequential Importance Sampling (SIS) (Handschin & Mayne, 1969)

- ▶ **Motivação:** em problemas de alta dimensão - $\mathbf{x} = (x_1, \dots, x_d)$, com d grande - é difícil encontrar uma boa distribuição g candidata.
- ▶ **Ideia:** construir $g(\mathbf{x})$ sequencialmente:

$$g(\mathbf{x}) = g_1(x_1)g_2(x_2|x_1) \cdots g_d(x_d|x, \dots, x_{d-1})$$

e decompor a distribuição alvo:

$$f(\mathbf{x}) = f(x_1)f(x_2|x_1) \cdots f(x_d|x, \dots, x_{d-1})$$

- ▶ Os pesos do IS tornam-se:

$$w(\mathbf{x}) = \frac{f(x_1)f(x_2|x_1) \cdots f(x_d|x, \dots, x_{d-1})}{g_1(x_1)g_2(x_2|x_1) \cdots g_d(x_d|x, \dots, x_{d-1})} \quad (7)$$

Sequential Importance Sampling (SIS)

- ▶ Seja $\mathbf{x}_t = (x_1, \dots, x_t)$. Então, os pesos do algoritmo IS podem ser calculados recursivamente:

$$\begin{aligned} w_t &= w_{t-1} \frac{f(x_t | \mathbf{x}_{t-1})}{g_t(x_t | \mathbf{x}_{t-1})} \\ &= w_{t-1} \frac{f(\mathbf{x}_t)}{f(\mathbf{x}_{t-1}) g_t(x_t | \mathbf{x}_{t-1})} \end{aligned} \quad (8)$$

- ▶ **Problema:** geralmente é difícil calcular a marginal $f(\mathbf{x}_t)$ para cada t ;
- ▶ **Ideia:** ache uma sequência de “distribuições auxiliares” $f_t(\mathbf{x}_t)$, $t = 1, \dots, d$ que aproxime as marginais $f(\mathbf{x}_t)$, tal que $f_d(\mathbf{x}) = f(\mathbf{x})$.

Algoritmo

Algorithm 3 Sequential Importance Sampling

```
1: for  $t = 1, \dots, T$  do
2:   Defina  $\tilde{f}_0^{(n)} = 1$  e  $\tilde{w}_0^{(n)} = 1$ 
3:   for  $n = 1, \dots, N$  do
4:     Amostre  $\mathbf{x}_t^{(n)} \stackrel{iid}{\sim} g_t(\mathbf{x}_t^{(n)} | \mathbf{x}_{t-1}^{(n)})$ 
5:     Faça  $\mathbf{x}_t^{(n)} = (\mathbf{x}_{t-1}^{(n)}, x_t^{(n)})$ 
6:     Atualize o peso não-normalizado:  $\tilde{w}_t^{(n)} = \tilde{w}_{t-1}^{(n)} \frac{\tilde{f}_t(\mathbf{x}_t^{(n)})}{\tilde{f}_t(\mathbf{x}_{t-1}^{(n)}) g_t(\mathbf{x}_t^{(n)} | \mathbf{x}_{t-1}^{(n)})}$ 
7:   Normalize os pesos:  $w_t^{(n)} = \frac{\tilde{w}_t^{(n)}}{\sum_{n=1}^N \tilde{w}_t^{(n)}}$ , para  $n = 1, \dots, N$ 
8: return  $(\mathbf{x}, \tilde{\mathbf{w}}, \mathbf{w})$ 
```

Aplicação 1: Self Avoid Walk (SAW)

- ▶ Aplicação proposta em (Zhou, 2022);
- ▶ Modelo simples para (bio)-polímeros;
- ▶ Considere o modelo de treliça (grade) 2-D;
- ▶ Um vetor $\mathbf{x}_T = (x_1, x_2, \dots, x_T)$ é um *self avoid walk* na treliça se:
 - $x_t = (a, b)$, onde a, b são inteiros;
 - $\text{distância}(x_t, x_{t+1}) = 1$;
 - $x_{t+1} \neq x_k, \forall k \leq t$.

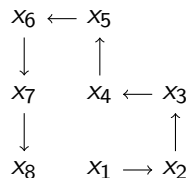


Figura 3: Exemplo de SAW. Neste exemplo, $x_1 = (0, 0), x_2 = (0, 1), \dots, x_8 = (0, -1)$.

Aplicação 1: Self Avoid Walk (SAW)

- ▶ Assuma que cada amostra SAW de tamanho T , $\mathbf{x}_T^{(n)}$ é gerado de forma equiprovável:
 - \mathbf{X}_T possui distribuição uniforme;
 - $f(\mathbf{x}_T) = \frac{1}{Z_T}$, onde Z_T é o número total possível de SAW de tamanho T .
- ▶ **Problemas:**
 - Como calcular Z_T ?
 - Como gerar amostras uniformes?

Aplicação 1: Self Avoid Walk (SAW)

► **Tentativa 1** (ingênua): *Random Walk*

- Comece em $x_1 = (0, 0)$;
- Ande para um vizinho aleatório (4 possíveis escolhas);
- Em $t \geq 2$, ande para qualquer uma das 3 posições $x_{t+1} \neq x_t$;
- Caso escolha alguma posição já ocupada anteriormente, reinicie em $(0, 0)$.

► **Problema:** Taxa de sucesso: $r = Z_T / (4 \times 3^{T-2})$ (Zhou, 2022);

- $T = 20, r \approx 21.6\%$;
- $T = 48, r \approx 0.79\%$;

Aplicação 1: Self Avoid Walk (SAW)

► **Tentativa 2** (um pouco menos ingênua):

- Comece em $x_1 = (0, 0)$;
- Ande para um vizinho aleatório que não está ocupado;

$$\mathbb{P}[x_{t+1} = (i', j') | x_1, \dots, x_t] = \frac{1}{n_t},$$

onde n_t é o número de vizinho de $x_t(i, j)$ não ocupados.

► **Problemas:**

- A geração de x_{t+1} depende de todo histórico x_t
 - Computacionalmente, não é um grande problema;
 - Em termos de inferência: processo não-Markoviano.
- Para $N > 9$, pode ficar preso e ter que reiniciar.
- As amostras geradas não possuem mesma probabilidade!

Aplicação 1: Self Avoid Walk (SAW)

- ▶ Amostras diferentes podem ser geradas com probabilidades diferentes, dependendo da sequência:

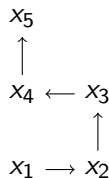


Figura 4: Neste exemplo, $\mathbb{P}[\mathbf{x}_t] = 1 \times \frac{1}{3} \times \frac{1}{3} \times \frac{1}{3} \times \frac{1}{2}$

$$x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow x_4 \rightarrow x_5$$

Figura 5: Neste exemplo, $\mathbb{P}[\mathbf{x}_t] = 1 \times \frac{1}{3} \times \frac{1}{3} \times \frac{1}{3} \times \frac{1}{3}$

Aplicação 1: Self Avoid Walk (SAW)

- ▶ Já sabemos simular amostras (ainda que não uniformes);
- ▶ Como amostrar uniformemente e contar Z_T ?
- ▶ **Tentativa 3** (ingênua): Método da força-bruta:
 - Para um dado T , simule uma amostra e armazene num conjunto (sem elementos repetidos);
 - Repita o procedimento por um número muito grande de passos;
 - Ao final, esperamos que o conjunto contenha todos os SAWs de dimensão T .
- ▶ **Problema:**
 - Não temos garantia de que de fato todas as possíveis amostras foram simuladas;
 - Z_T cresce exponencialmente com T , requerendo um número ainda maior de iterações para tentar obter o conjunto inteiro de amostras.

Aplicação 1: Self Avoid Walk (SAW)

► Solução por Sequential Monte Carlo:

- Distribuição alvo: $f_t(\mathbf{x}_t) \propto 1$;
- Distribuição proposta: $g_t(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathbb{P}[\mathbf{x}_t|\mathbf{x}_{t-1}] = \frac{1}{n_{t-1}}$;
- Atualização dos pesos: $w_t = w_{t-1} n_{t-1}$

► Simulação:

- Usamos $N = 100.000$ amostras para estimar Z_T , para $T = 3, \dots, 20$. Os resultados são mostrados no gráfico da fig. 6;
- Estimamos também Z_T pelo método da bruta-força até $T = 10$ (próximo slide);

Aplicação 1: Self Avoid Walk (SAW)

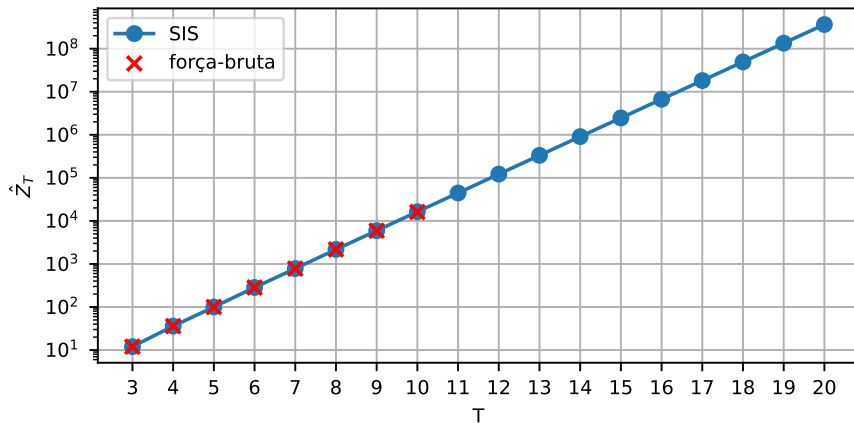


Figura 6: Estimativa de Z_T pelos métodos da força-bruta e Sequential Importance Sampling

Aplicação 1: Self Avoid Walk (SAW)

- ▶ Os valores de Z_T estimados por SIS foram bastante próximos ou iguais aos valores estimados por força bruta;
- ▶ Z_T cresce exponencialmente com T ;
- ▶ Observe que, para $T = 12$, Z_T (estimado por SIS) já é maior que 100.000;
- ▶ Para $T = 20$, $\hat{Z}_T > 10^8$. Ou seja, se quiséssemos simular por força-bruta, teríamos que gerar mais de 100 milhões de amostras, o que torna o método da força bruta proibitivo computacionalmente (não factível de execução);
- ▶ Usando reamostragem das amostras simuladas, com probabilidade dada pelos *importance weights*, poderíamos obter uma amostra uniforme, aproximada, de $f(\mathbf{x})$.

Aplicação 2: Modelo de Volatilidade Estocástica

- ▶ Modelo simples de volatilidade estocástica (Fearnhead, 2012)

$$x_t = \mathcal{N}(\phi x_{t-1}, \sigma^2) = r(x_t | x_{t-1}) \quad (9)$$

$$y_t = \mathcal{N}(0, e^{(\gamma + x_t)}) = s(y_t | x_t) \quad (10)$$

- ▶ y_t : retornos (**variável observada**)
- ▶ x_t : volatilidade estocástica (**variável latente**):
 - Medida de variabilidade dos preços;
- ▶ **Objetivo da aplicação de SIS:**
 - Dado y_t , estimar (“filtrar”), de forma *online*, x_t ;

Aplicação 2: Modelo de Volatilidade Estocástica

► **Retornos** (y_t):

- Preço do ativo no período t : P_t
- Retorno: $r_t = \log P_t - \log P_{t-1}$

► **Volatilidade** (σ_t^2):

- $\sigma_t^2 = \mathbb{E}[r_t^2 | \mathcal{F}_{t-1}]$;
 - \mathcal{F}_{t-1} pode ser interpretado como todas as informações que se tem até $t - 1$;

► **Volatilidade estocástica** (x_t):

- Transformação não linear $x_t = \log(\sigma_t^2)$

Aplicação 2: Modelo de Volatilidade Estocástica

► Ideias do modelo (Morettin, 2017):

- A volatilidade presente depende de seus valores passados, mas é independente dos retornos passados;
- Reproduz o fenômeno observado (fato estilizado) de *agrupamentos de volatilidade* em séries temporais financeiras;
- Períodos de alta (baixa) volatilidade tendem a ser seguidos por períodos de alta (baixa) volatilidade;

Aplicação 2: Modelo de Volatilidade Estocástica

Aplicação de Sequential Importance Sampling

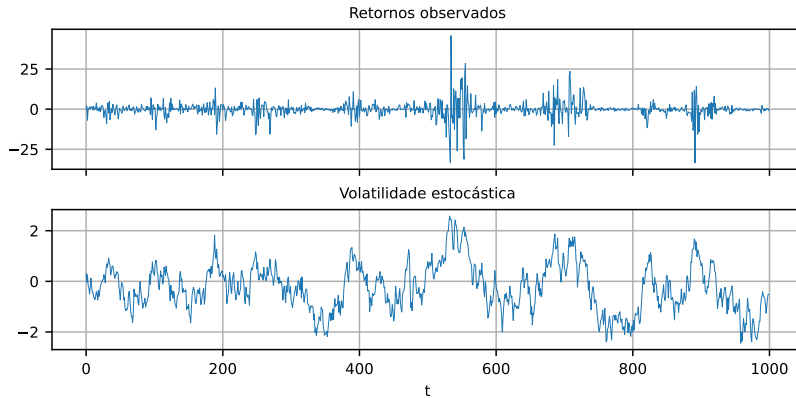


Figura 7: Modelo de volatilidade estocástica simulado.

Parâmetros: $\phi = 0.95$, $\sigma = \sqrt{1 - \phi^2}$, $\gamma = 1$

Aplicação 2: Modelo de Volatilidade Estocástica

Aplicação de Sequential Importance Sampling

- ▶ Para aplicarmos SIS, primeiramente precisamos definir as funções alvo (f) e proposta (g);
- ▶ No caso do SSM, queremos estimar o estado latente conhecendo a variável observada:
 - $p(\mathbf{x}_t | \mathbf{y}_t)$
- ▶ Definimos então a seguinte função alvo:

- $\tilde{f}_t(\mathbf{x}_t) = p(\mathbf{x}_t | \mathbf{y}_t) = \frac{p(\mathbf{x}_t, \mathbf{y}_t)}{p(\mathbf{y}_t)}$

- $f_t(\mathbf{x}_t) = p(\mathbf{x}_t, \mathbf{y}_t)$

$$f_t(\mathbf{x}_t) = p(x_1) \prod_{k=2}^t r(x_k | x_{k-1}) \prod_{k=1}^t s(y_k | x_k) \quad (11)$$

Aplicação 2: Modelo de Volatilidade Estocástica

Aplicação de Sequential Importance Sampling

- ▶ E a seguinte função proposta (função de transição de estados do SSM):

$$g_t(x_t | \mathbf{x}_{t-1}) = r(x_t | x_{t-1}) \quad (12)$$

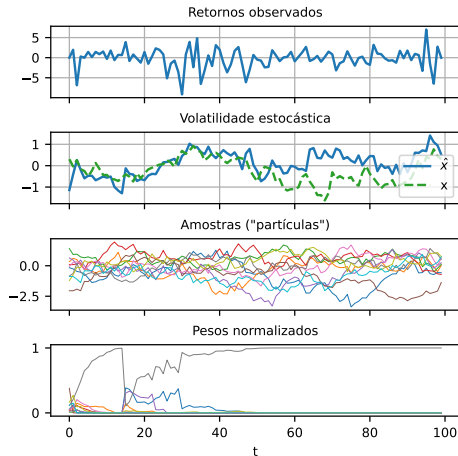
- ▶ A equação de atualização dos pesos fica então:

$$\tilde{w}_t = \tilde{w}_{t-1} \times s(y_t | x_t) \quad (13)$$

- ▶ Realizamos a aplicação do SIS considerando:
 - Número de amostras (partículas): $N = 10$;

Aplicação 2: Modelo de Volatilidade Estocástica

Aplicação de Sequential Importance Sampling



Aplicação 2: Modelo de Volatilidade Estocástica

Aplicação de Sequential Importance Sampling

- ▶ No gráfico acima, temos no painel superior a variável observada (retornos) e logo abaixo a volatilidade estocástica real (x) e estimada (\hat{x}). Plotamos também as amostras da distribuição proposta (partículas) e os respectivos pesos normalizados;
- ▶ No painel inferior, podemos observar o seguinte comportamento dos pesos: com poucas iterações, os pesos evoluem rapidamente para 0, com exceção de uma única amostra, a qual o peso converge para 1;
- ▶ Isto acarreta uma deterioração da volatilidade estocástica estimada \hat{x} para instantes após $t = 40$;
- ▶ Ao aumentarmos o número de amostras (por exemplo, $N = 100$), o problema persiste.

Problema da degeneração dos pesos

- ▶ Infelizmente, este é um fenômeno que ocorre frequentemente no SIS e é conhecido como o **problema da degeneração dos pesos**;
- ▶ É a principal limitação para aplicação do SIS em problemas práticos;
- ▶ O algoritmo **Sequential Monte Carlo**, apresentado a seguir, ameniza o problema e consegue solução satisfatória em vários casos, mas não em todos. Segundo Naesseth et al. (2019, pg. 25) a degeneração dos pesos ainda é um **problema em aberto**.

Sequential Monte Carlo (SMC)

Sequential Monte Carlo (SMC)

- ▶ Os métodos de Monte Carlos sequenciais (SMC) atacam o problema de degeneração dos pesos escolhendo uma proposta g_t que considera as informações contidas em \hat{f}_{t-1} , a distribuição alvo estimada no instante anterior;
- ▶ Isto é feito da seguinte forma:
 - **Passo 1 - Reamostragem:** Usando o conjunto atual de pesos e amostras $\{(\mathbf{x}_{t-1}^{(n)}, w_{t-1}^{(n)})\}_{n=1}^N$, reamostrar com reposição N amostras e substituir $\mathbf{x}_{t-1}^{(n)}$ por estas novas amostras;
 - Neste passo, estamos fazendo $\mathbf{x}_{t-1}^{(n)} \sim \hat{f}_{t-1}(\mathbf{x}_{t-1})$;
 - **Passo 2 - Propagação:** Amostrar $x_t^{(n)}$ da distribuição proposta $g_t(x_t | \mathbf{x}_{t-1}^{(n)})$;
 - **Passo 3 - Concatenação:** Fazer $\mathbf{x}_t^{(n)} = (x_t^{(n)}, \mathbf{x}_{t-1}^{(n)})$;

Sequential Monte Carlo (SMC)

- ▶ Diferentemente do SIS, agora a expressão para atualização dos pesos não leva mais em conta (diretamente) o peso no instante anterior:
 - g_t já incorpora estas informações

$$\tilde{w}_t^{(n)} = \frac{\tilde{f}_t(\mathbf{x}_t^{(n)})}{\tilde{f}_t(\mathbf{x}_{t-1}^{(n)})g_t(x_t^{(n)}|\mathbf{x}_{t-1}^{(n)})} \quad (14)$$

- ▶ A etapa de normalização dos pesos é feita como anteriormente;
- ▶ No SMC, devido ao passo de reamostragem, $\mathbf{x}_t^{(n)}$ já são amostras estimadas de $f(\cdot)$. Então, para calcularmos valores esperados, usamos uma média simples:

$$\mathbb{E}_f[h(\mathbf{X})] = \frac{1}{N} \sum_{n=1}^N h(\mathbf{x}^{(n)}) \quad (15)$$

Algoritmo

Algorithm 4 Sequential Monte Carlo

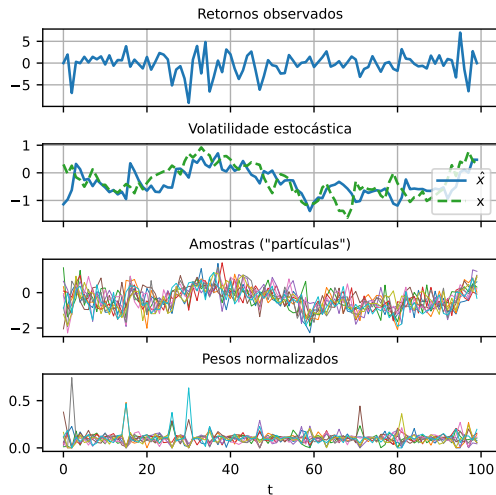
```

1: for  $t = 1, \dots, T$  do
2:   Defina  $\tilde{f}_0^{(n)} = 1$  e  $\tilde{w}_0^{(n)} = 1$ 
3:   for  $n = 1, \dots, N$  do
4:     Reamostragem: Reamostre  $\mathbf{x}_{t-1}^{(n)}$  à partir de  $\{(\mathbf{x}_{t-1}^{(n)}, w_{t-1}^{(n)})\}_{n=1}^N$ 
5:     Propagação: Amostre  $x_t^{(n)} \stackrel{iid}{\sim} g_t(x_t^{(n)} | \mathbf{x}_{t-1}^{(n)})$ 
6:     Concatenação: Faça  $\mathbf{x}_t^{(n)} = (\mathbf{x}_{t-1}^{(n)}, x_t^{(n)})$ 
7:     Atualize o peso não-normalizado:  $\tilde{w}_t^{(n)} = \frac{\tilde{f}_t(\mathbf{x}_t^{(n)})}{\tilde{f}_t(\mathbf{x}_{t-1}^{(n)})g_t(x_t^{(n)} | \mathbf{x}_{t-1}^{(n)})}$ 
8:   Normalize os pesos:  $w_t^{(n)} = \frac{\tilde{w}_t^{(n)}}{\sum_{n=1}^N \tilde{w}_t^{(n)}}$ , para  $n = 1, \dots, N$ 
9: return  $(\mathbf{x}, \mathbf{w})$ 

```

Aplicação: Modelo de Volatilidade Estocástica

- ▶ Mesmo modelo de volatilidade estocástica estudado anteriormente com SIS;
 - $N = 10$ amostras;
- ▶ Com SMC: problema da degeneração dos pesos não ocorreu;
- ▶ Na figura a seguir, temos o mesmo experimento, porém com uma janela de $T = 1000$ instantes de tempo.



Aplicação: Modelo de Volatilidade Estocástica

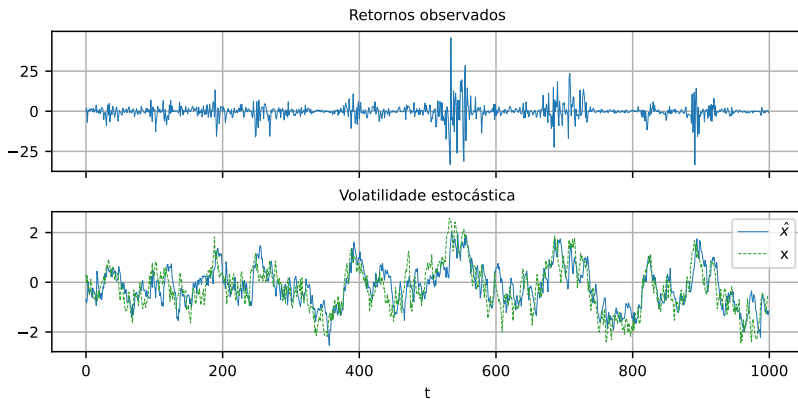


Figura 8: Modelo de volatilidade estocástica: retornos observados e volatilidades estocásticas real (latente, x_t) e inferida (\hat{x}_t) pelo método SMC. Podemos observar que o método conseguiu inferir de forma satisfatória a variável latente.

Extensões

- ▶ Existem na literatura diversas extensões e melhorias do método SMC básica apresentamos e implementamos neste trabalho. Comentamos brevemente algumas, discutidas em (Naesseth et al., 2019);
- ▶ A etapa de reamostragem aumenta a variância das estimativas no SMC, incluindo os pesos. Alternativas:
 - Reamostragem com menor variância: estratificada, sistemática;
 - Reamostragem adaptativa monitorando o Tamanho Efetivo da Amostra;
- ▶ Escolha otimizada da função proposta:
 - SMC Adaptativo;
 - SMC Variacional;
- ▶ Combinação de SMC com outros métodos MCMC.

Conclusões






Conclusões

- ▶ Neste trabalho, estudamos e implementamos exemplos e aplicações simples para os métodos de **Importance Sampling, Sequential Importance Sampling e Sequential Monte Carlo**;
- ▶ As aplicações estudadas permitiram obtermos uma compreensão introdutória da área, bem como um vislumbre das potencialidades e limitações dos métodos;
- ▶ Concluímos que os métodos de Monte Carlo sequenciais são **uma poderosa ferramenta para inferência aproximada**; permitem, por exemplo, inferência em sistemas não lineares;
- ▶ Também constatamos que trata-se de uma ampla área pesquisa, com bastante teoria já desenvolvida; mas que ainda é bastante ativa e apresenta diversos desafios e oportunidades.

Implementação

- ▶ Implementamos o exemplo de IS em R e as aplicações de SIS e SMC em Python;
- ▶ O código-fonte dos exemplos e aplicações implementadas encontra-se em <https://github.com/cleitonmoya/smc>.

Referências I

-  Chopin, N., & Papaspiliopoulos, O. (2020). *An introduction to sequential Monte Carlo* (Vol. 4). Springer.
-  Fearnhead, P. (2012). GTP 2012: Modern Computational Statistics (Alternatives to MCMC). <https://www.maths.lancs.ac.uk/~fearnhea/GTP>
-  Gordon, N. J., Salmond, D. J., & Smith, A. F. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEEE Proceedings F (Radar and Signal Processing)*, 140(2), 107–113.
-  Handschin, J. E., & Mayne, D. Q. (1969). Monte Carlo techniques to estimate the conditional expectation in multi-stage non-linear filtering. *International journal of control*, 9(5), 547–559.
-  Morettin, P. A. (2017). *Econometria financeira: um curso em séries temporais financeiras* (3ª ed.). Editora Blucher.

Referências II



Naesseth, C. A., Lindsten, F., Schön, T. B., et al. (2019). Elements of sequential monte carlo. *Foundations and Trends® in Machine Learning*, 12(3), 307–392.



Zhou, Q. (2022). Stats 102C - Introduction to Monte Carlo Methods.
<http://www.stat.ucla.edu/~zhou/courses/Stats102C-IS.pdf>