Aula 4

Os Métodos Monte Carlo

Introdução

Em estatística, denomina-se "Monte Carlo"qualquer método que tenha como característica o uso de diversas experiências práticas (amostragens) para simular um valor probabilístico teórico que, por quaisquer motivos, não é possível calcular diretamente. É uma família de algoritmos muito atraente, considerando que, na vida real frequentemente desejamos entender como funcionam as probabilidades por trás de diversos eventos (clima, mercado, etc), e temos acesso a vários dados passados que assumimos como amostras deles (dias que choveram em uma certa estação, vezes que uma ação subiu dada uma atitude da empresa, etc).

No contexto de Reinforcement Learning, nossa ideia na prática será utilizar uma série de simulações do agente em nosso ambiente, seguindo uma certa política, e utilizá-las como base para tentar estimar as funções ação-valor de nossos estados nessa situação. No contexto do GPI, isso significa que a parte de *policy evaluation* será avaliada dessa forma, e a *policy improvement*, por sua vez, pode ser igual ao método da DP ou um tanto diferente (dependendo do contexto), mas mantém-se relativamente independente.

Os métodos Monte Carlo apresentam diferenças marcantes dos de DP, sendo vantajosos em diversas situações. Isso ficará mais claro conforme você ler esta apostila; por hora, é possível citar:

- **Aprendizado por experiência/sem modelo:** para muitos problemas práticos, não sabemos as consequências de nossas ações com precisão numérica, ou seja, não temos um modelo do ambiente ($p(s', r|s, a)$), e portanto usar DP é impossível.

- **Foco em estados frequentes:** como veremos a seguir, é possível simular episódios com nosso agente em alguns estados específicos que desejarmos, permitindo que treinemos mais o agente em cenários mais recorrentes, melhor utilizando tempo e recursos computacionais.

- **Facilidade em simular o episódio:** em contraste com a dificuldade de obter um modelo, frequentemente é fácil simular nosso episódio para que o agente aprenda

Monte Carlo Prediction

A princípio, vamos considerar o caso de tentar estimar a função valor $v_\pi(s)$ para cada estado $s \in S$, seguindo uma política π. É possível fazer o mesmo para $q_\pi(s, a)$ (subseção seguinte), o que é mais comum, já que isso permite que façamos *policy improvement* sem ter nenhum modelo $p(s', r|a, s)$ do ambiente. De qualquer modo, é interessante analisar o uso de Monte Carlo nesse contexto inicial, a fim de aprender o básico do método.

Vamos definir mais rigorosamente do que se trata esse "aprendizado por experiência". Simularemos vários episódios com nosso agente (assumiremos MDPs episódicos), e manteremos guardados o valor de cada retorno para cada estado; atualizaremos sucessivamente, para cada episódio, o valor de $V_\pi(s)$ como a média dos retornos obtidos para cada estado s. Pela lei dos números grandes, nossa estimativa $V_\pi(s)$ estará cada vez mais correta quanto maior o número n de episódios, isto é: $\lim_{n\to\infty} V_\pi(s) = v_\pi(s)$. O pseudocódigo disso é:

Algorithm 1 Monte Carlo Prediction

Inicializar:
$\pi \leftarrow$ política a ser simulada
$V \leftarrow$ valores arbitrários
$G(s) \leftarrow 0$; lista vazia de retornos, $s \in S$
for $i = 1$ até n **do**
    Gerar episódio com π
    for cada s do episódio gerado **do**
        Calcular G_s
        Anexar G_s a $G(s)$
        $V \leftarrow \frac{G(s)}{n}$
    end for
end for

Exemplo: BlackJack

Como os próprios autores do livro base desse curso (Sutton e Barto) afirmam: "o uso dos métodos MC é melhor ilustrado através de exemplos". Portanto, imagine o seguinte cenário: deseja-se simular um jogo de BlackJack entre o agente e um *dealer* (se você não conhece o jogo, leia os primeiros 2 parágrafos desse link, assim como a seção de jogadas). Para cada política que quisermos, teremos um MDP com os seguintes elementos:

- $r \in (+1, 0, -1)$ para ganhar, perder ou empatar;
- $\gamma = 1 \rightarrow R = G$ (s/descontos, ou seja, as recompensas serão igual aos retornos)
- $A(s) = (hit, stick)$
- S definido por 3 fatores: soma das cartas (até 21), qual carta o *dealer* está exibindo, e um *ace* usável ou não.

Com 2 adendos: a carta *ace* pode estar em mãos do agente, e só será "usável"se puder ser utilizada com valor +11, pois do contrário ela causará *bust*; o *deck* de cartas sempre tem todas as cartas do jogo (amostragem c/reposição). Nas condições descritas, seria difícil determinar $p(s', r|s, a)$, mas é fácil simular um agente nessa situação. Por exemplo, para a política π de *hit* caso soma<20, do contrário *stand*, teremos o seguinte $V_\pi(s)$:

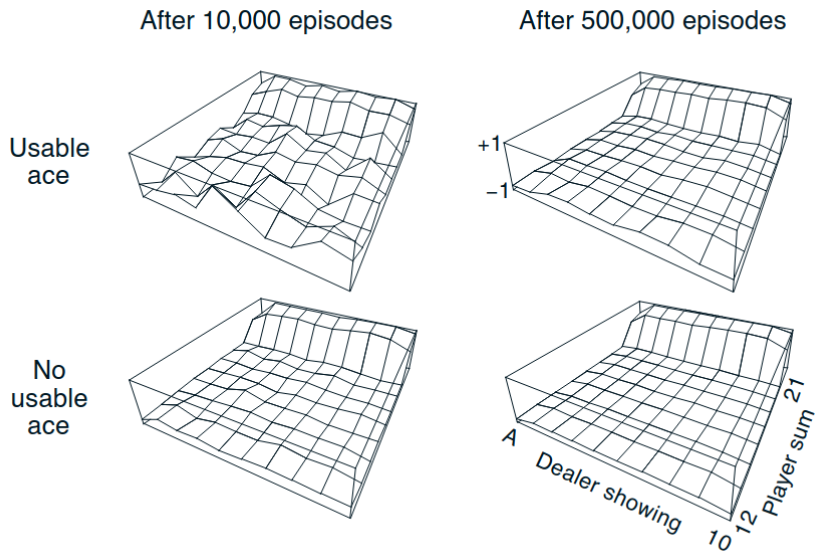

Figura 1: Evolução de $V_\pi(s)$, com cada quadradinho sendo um estado, que vale entre $-1, +1$; o exemplo e a imagem são do livro base do curso

Monte Carlo para Valores de Ação e Controle

Por que estimar valores de ação?

Sem um modelo do ambiente, apenas valores de estado $v^\pi(s)$ não bastam para decidir ações. Precisamos estimar *valores de ação*

$$q^\pi(s, a) \doteq \mathbb{E}_\pi[G_t \mid S_t = s, A_t = a],$$

pois uma política pode então ser escolhida tomando, em cada estado, a ação com maior valor estimado.

Estimativa Monte Carlo de q^π

Os métodos Monte Carlo (MC) para q^π são análogos aos de v^π, trocando "visitas ao estado" por "visitas ao par estado–ação (s, a)":

- **Primeira-visita (first-visit):** média dos retornos que seguem a *primeira* ocorrência de (s, a) em cada episódio.
- **Todas-as-visitas (every-visit):** média dos retornos após *todas* as ocorrências de (s, a) em cada episódio.

Se cada par (s, a) for visitado infinitas vezes, ambas as variantes convergem ao valor esperado verdadeiro.

Questão central: exploração. Com uma política determinística, observamos retornos para apenas uma ação por estado; os demais $Q(s, a)$ não melhoram. Para garantir cobertura de todos os pares (s, a), há duas abordagens:

1. **Exploring starts:** cada episódio inicia em algum (s, a) com probabilidade > 0 para *todos* os pares. Assim, todos são visitados infinitamente na soma de episódios.

2. **Políticas estocásticas:** restringir-se a políticas que escolhem *todas* as ações com probabilidade > 0 em cada estado (e.g., políticas *soft*/ϵ-soft).

Na prática, *exploring starts* costuma ser uma hipótese teórica; políticas estocásticas são a alternativa comum.

Controle com Métodos Monte Carlo

A Estrutura do GPI Aplicada ao Monte Carlo

Agora que sabemos como avaliar uma política, o próximo passo é descobrir como melhorá-la para encontrar a política ótima. Para isso, vamos usar a mesma estrutura da **Iteração de Política Generalizada (GPI)** que já conhecemos, adaptada para os métodos Monte Carlo. O ciclo é composto pelos dois processos já familiares:

- **Avaliação de Política (E):** Usando a política atual, π_k, rodamos vários episódios no ambiente. Ao final, calculamos a média dos retornos para cada par estado-ação (s, a) para estimar a função de valor-ação, $q_{\pi k}$.

- **Melhoria de Política (I):** Com a função $q_{\pi k}$ em mãos, atualizamos a política para que ela se torne "gananciosa"(*greedy*) em relação a esses valores. A nova política, π_{k+1}, simplesmente escolherá a ação com o maior valor q para cada estado.

O processo completo segue o fluxo $\pi_0 \xrightarrow{E} q_{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} q_{\pi_1} \xrightarrow{I} \cdots \rightarrow \pi_*$. Uma grande vantagem aqui é que, ao trabalharmos com a função de valor-ação $q(s, a)$, **não precisamos de um modelo do ambiente** para realizar a melhoria da política, pois o valor de cada ação já está explicitamente estimado.

Desafios Práticos e Soluções

A abordagem descrita acima funciona na teoria, mas se baseia em duas suposições pouco realistas para garantir a convergência:

1. **A avaliação exige infinitos episódios:** Para que $q_{\pi k}$ seja a estimativa exata do valor da política, precisaríamos de uma quantidade infinita de episódios.

2. **O ambiente permite inícios exploratórios (*exploring starts*):** Para garantir que todos os pares (s, a) sejam visitados, o método assume que podemos iniciar um episódio de qualquer estado e tomando qualquer ação inicial.

Felizmente, podemos adaptar o algoritmo para contornar esses problemas.

Para o primeiro problema, a solução é **não esperar a avaliação convergir completamente**. Assim como no GPI, podemos intercalar os processos de forma mais fina. A abordagem mais natural para os métodos Monte Carlo é alternar a cada episódio:

- Roda-se um episódio completo.
- Usa-se os retornos observados para dar um pequeno passo na atualização dos valores de q (um passo de avaliação).
- A política é então imediatamente melhorada para os estados que foram visitados naquele episódio (um passo de melhoria).

Dessa forma, a função de valor e a política evoluem juntas gradualmente.
O segundo problema, que envolve a necessidade de exploring starts, é mais complexo e será abordado em seções futuras, onde veremos técnicas para garantir a exploração de forma mais natural.

Exemplo: Resolvendo o Blackjack

Para ilustrar a força desse método, o algoritmo Monte Carlo ES (Exploring Starts) foi aplicado ao jogo de Blackjack. Como o ambiente era simulado, foi possível forçar os inícios exploratórios para garantir a visita a todas as situações de jogo. O resultado foi a descoberta da política ótima para o Blackjack, também conhecida como a "estratégia básica"do jogo, mostrando que o método funciona na prática.

Métodos On-Policy em Monte Carlo

Até agora vimos como os métodos de Monte Carlo permitem realizar predição e controle a partir de episódios gerados sob uma política. Entretanto, para que o controle seja de fato possível, é necessário lidar com algumas dificuldades práticas relacionadas à **exploração**.

O problema de episódios infinitos

Para que as estimativas de valor $q_\pi(s, a)$ sejam corretas, cada par estado-ação (s, a) precisa ser visitado um número suficientemente grande de vezes. Na prática, isso significa que o algoritmo depende de um *número infinito de episódios* para garantir convergência. Em problemas reais, não é viável esperar visitas infinitas; portanto, precisamos de mecanismos que incentivem a exploração de todas as ações relevantes em todos os estados.

O problema de Exploring Starts

Uma solução teórica apresentada inicialmente é a hipótese de **Exploring Starts**: supor que cada episódio começa em um estado-ação aleatório com probabilidade não nula. Isso garante que todos os pares (s, a) eventualmente serão explorados. Contudo, essa hipótese é irrealista na maioria das aplicações, pois dificilmente temos controle sobre o estado inicial do ambiente.

Exploração e políticas ϵ-gananciosas

Sem exploração adequada, o agente pode entrar em um ciclo vicioso: uma vez que uma ação não é escolhida inicialmente, ela nunca é visitada; se nunca é visitada, nenhum dado é coletado sobre ela; sem dados, o agente nunca poderá descobrir que essa ação poderia ser melhor em certas situações. Em outras palavras, ações podem ser simplesmente **esquecidas** pelo algoritmo, ficando permanentemente de fora da política aprendida.

Se a política fosse puramente gananciosa (*greedy*) em relação às estimativas atuais de valor, o agente exploraria apenas o que já parece ser bom, correndo o risco de nunca descobrir alternativas superiores. Isso torna o aprendizado incompleto e potencialmente preso em soluções subótimas.

A solução prática é utilizar uma política ϵ**-gananciosa** (ϵ-*greedy*):

- Com probabilidade $1 - \epsilon$, a ação escolhida é aquela que maximiza o valor estimado $q(s, a)$ (explotação).
- Com probabilidade ϵ, a ação escolhida é aleatória entre as disponíveis (exploração).

Dessa forma, garantimos que todas as ações tenham alguma chance de serem escolhidas, o que impede que ações potencialmente boas sejam esquecidas. Além disso, o ϵ-greedy mantém o equilíbrio necessário: o agente continua explorando novas possibilidades, mas ainda prioriza o uso do conhecimento já adquirido.

Ou seja, o ϵ-greedy permite **explorar enquanto explotamos**, mantendo o aprendizado ativo sem abrir mão de utilizar o que já foi aprendido.

Algoritmo Monte Carlo On-Policy com ϵ-greedy

A seguir, apresentamos o algoritmo de controle Monte Carlo **on-policy** com política ϵ-gananciosa, conforme descrito em Sutton & Barto (2018):

Algorithm 2 Controle Monte Carlo On-Policy com política ϵ-soft

Parâmetro do algoritmo: pequeno $\epsilon > 0$
Inicializar π: política arbitrária ϵ-soft
Inicializar $Q(s, a)$ arbitrário e $Returns(s, a)$ lista vazia para todo s, a
repeat ▷ para cada episódio
    Gerar episódio $(S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T)$ seguindo π
    $G \leftarrow 0$
    for $t = T - 1, T - 2, \ldots, 0$ **do**
        $G \leftarrow G + R_{t+1}$
        if (S_t, A_t) não aparece em $(S_0, A_0), \ldots, (S_{t-1}, A_{t-1})$ **then**
            Acrescentar G em $Returns(S_t, A_t)$
            $Q(S_t, A_t) \leftarrow$ média de $Returns(S_t, A_t)$
            $A^* \leftarrow \arg\max_a Q(S_t, a)$
            for cada $a \in A(S_t)$ **do**

Atualizar política:

$$\pi(a|S_t) \leftarrow \begin{cases} 1 - \epsilon + \dfrac{\epsilon}{|A(S_t)|}, & \text{se } a = A^* \\ \dfrac{\epsilon}{|A(S_t)|}, & \text{se } a \neq A^* \end{cases}$$

            end for
        end if
    end for
until para sempre

Esse algoritmo ilustra a essência do método *on-policy*: a política que gera os episódios é a mesma que está sendo avaliada e melhorada. A combinação da estimativa de $Q(s, a)$ com a atualização ϵ-gananciosa garante que a política não congele prematuramente em soluções subótimas, mantendo um equilíbrio entre **exploração** e **explotação**.

Métodos de Controle Off-Policy em Monte Carlo

A Abordagem Off-Policy

Até agora, os métodos que vimos são do tipo **on-policy**. Eles avaliam e melhoram a mesma política que está sendo usada para gerar os dados (os episódios). Esta abordagem representa uma troca: em vez de aprender a política ideal, o agente aprende sobre a melhor política possível que ainda mantém um grau de exploração.

Disso, nos questionamos: como podemos aprender sobre uma política ótima enquanto seguimos uma política exploratória diferente? A resposta está nos métodos **off-policy**.

Em um método off-policy (ou "fora da política"), separamos as funções de exploração e de aprendizado. Para isso, utilizamos duas políticas distintas:

- **Política-alvo (π):** É a política que queremos aprender e otimizar. Geralmente, esta é a política gananciosa (*greedy*) em relação às estimativas atuais de valor, sem exploração.
- **Política de comportamento (b):** É a política que o agente de fato segue para gerar os episódios. Ela precisa ser mais exploratória (ex: ϵ-gananciosa) para garantir que continue visitando vários estados e ações.

O objetivo é estimar a função de valor $v_\pi(s)$ ou $q_\pi(s, a)$ da política-alvo π, usando retornos G_t que foram obtidos seguindo a política de comportamento b. A abordagem on-policy é, na verdade, um caso especial da off-policy, onde π e b são a mesma política.

Amostragem por Importância (Importance Sampling)

Se o agente segue a política b, os retornos que ele observa refletem a qualidade de b, não de π. De que maneira podemos usar essa experiência para aprender sobre π?

A solução é a **Amostragem por Importância** (*Importance Sampling*), uma técnica para estimar valores de uma distribuição utilizando amostras de outra. A ideia é ponderar cada retorno obtido, ajustando sua "importância"para que ele se assemelhe a um retorno que possa ter sido gerado pela política-alvo π.

Condição Essencial: Cobertura (Coverage)

Para que isso funcione, precisamos garantir que a política de comportamento b explore o suficiente. O pressuposto de **cobertura (assumption of coverage)** exige que toda ação que π pode tomar, b também deve poder tomar com alguma probabilidade. Formalmente, é necessário que $\pi(a|s) > 0$ implique $b(a|s) > 0$.

Razão de Amostragem por Importância (ρ)

O fator de ponderação é chamado de **razão de amostragem por importância**, denotado por ρ. Ele mede a probabilidade relativa de uma trajetória ocorrer sob π em comparação com b.

$$\rho_{t:T-1} \doteq \frac{\prod_{k=t}^{T-1} \pi(A_k|S_k)p(S_{k+1}|S_k, A_k)}{\prod_{k=t}^{T-1} b(A_k|S_k)p(S_{k+1}|S_k, A_k)} = \prod_{k=t}^{T-1} \frac{\pi(A_k|S_k)}{b(A_k|S_k)}$$

Como a dinâmica do ambiente (p) é a mesma para ambas as políticas, os termos se cancelam. Isso é uma grande vantagem, pois o método se torna **livre de modelo (model-free)**.

O valor de ρ nos diz o quão relevante é uma experiência gerada por b para aprender sobre π:

- $\rho > 1$: A trajetória era mais provável de acontecer sob a política-alvo π. A experiência é muito relevante.

- $\rho < 1$: A trajetória era menos provável de acontecer sob π.
- $\rho = 0$: A política-alvo π jamais teria seguido essa trajetória. A experiência é irrelevante e será descartada.

Com esta razão, podemos corrigir o valor esperado do retorno, de modo que a média dos retornos ponderados convirja para o valor da política-alvo:

$$\mathbb{E}[\rho_{t:T-1} G_t | S_t = s] = v_\pi(s)$$

O Problema do Viés e da Variância

Existem duas formas principais de aplicar a amostragem por importância, e elas apresentam um trade-off entre viés (bias) e variância (variance).

Amostragem por Importância Ordinária (Ordinary)

Calcula uma média simples dos retornos ponderados.

$$V(s) \doteq \frac{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)-1} G_t}{|\mathcal{T}(s)|}$$

- **Vantagem:** É **não viciada (unbiased)**. Em média, a estimativa converge para o valor verdadeiro $v_\pi(s)$.
- **Desvantagem:** Possui **alta variância**. Se uma razão ρ for muito grande, um único episódio pode dominar a média, tornando o aprendizado instável e lento.

Amostragem por Importância Ponderada (Weighted)

Calcula uma média ponderada dos retornos.

$$V(s) \doteq \frac{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)-1} G_t}{\sum_{t \in \mathcal{T}(s)} \rho_{t:T(t)-1}}$$

- **Vantagem:** Tem uma **variância muito menor** e é muito mais estável na prática, sendo a escolha padrão.
- **Desvantagem:** É **viciada (biased)**. As estimativas iniciais são enviesadas em direção aos retornos da política de comportamento b. No entanto, esse viés desaparece à medida que mais dados são coletados.

O gráfico abaixo ilustra como a Amostragem Ponderada (vermelho) converge de forma mais estável que a Ordinária (verde).

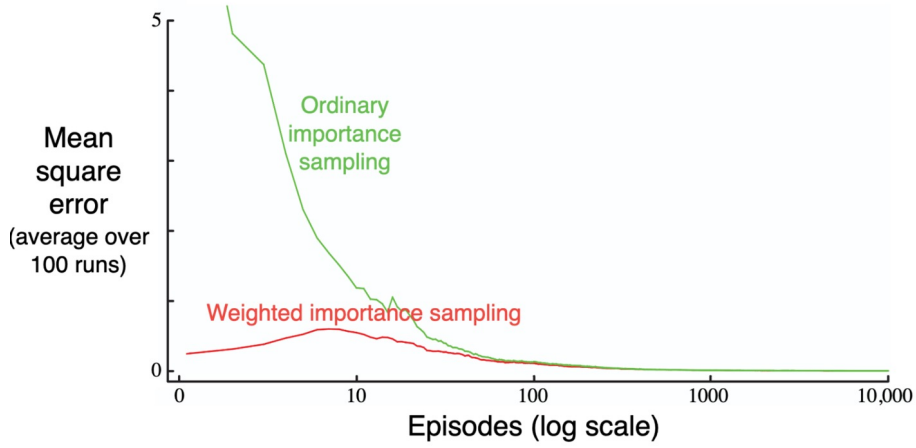

Figura 2: Comparação entre Amostragem por Importância Ordinária e Ponderada. A Ponderada apresenta menor erro quadrático médio e maior estabilidade.

Implementação Incremental

Para evitar a ineficiência de recalcular a média completa a cada episódio, usamos uma abordagem incremental. Mantemos uma soma cumulativa dos pesos (C) e atualizamos a média (V) a cada novo retorno (G) e peso (W).

A estimativa V_n é a média ponderada dos retornos:

$$V_n \doteq \frac{\sum_{k=1}^{n-1} W_k G_k}{\sum_{k=1}^{n-1} W_k}$$

A atualização incremental é:

$$V_{n+1} \doteq V_n + \frac{W_n}{C_n}[G_n - V_n], \quad \text{onde} \quad C_{n+1} \doteq C_n + W_{n+1}$$

Algoritmo de Controle Monte Carlo Off-Policy

Agora, podemos criar um algoritmo de controle off-policy completo. O objetivo é aprender os valores de ação ótimos, q_*, usando uma política-alvo π gananciosa e uma política de comportamento b exploratória.

Uma característica do algoritmo é aprender a partir do fim dos episódios. Ele processa a trajetória de trás para frente e para assim que encontra uma ação que a política-alvo (gananciosa) não teria tomado. A partir desse ponto, a trajetória não é mais representativa da política-alvo e o aprendizado para aquele episódio é interrompido.

Algorithm 3 Controle Monte Carlo Off-Policy para estimar $\pi \approx \pi_*$

Inicializar, para todo $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$:
    $Q(s, a) \in \mathbf{R}$ (arbitrariamente)
    $C(s, a) \leftarrow 0$
    $\pi(s) \leftarrow \arg\max_a Q(s, a)$ ▷ Política-alvo gananciosa
loop ▷ Para cada episódio
    $b \leftarrow$ qualquer política ϵ-soft ▷ Política de comportamento
    Gerar episódio $(S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T)$ seguindo b
    $G \leftarrow 0$
    $W \leftarrow 1$
    for $t = T - 1, T - 2, \ldots, 0$ **do**
        $G \leftarrow \gamma G + R_{t+1}$
        $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
        $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W}{C(S_t, A_t)}[G - Q(S_t, A_t)]$
        $\pi(S_t) \leftarrow \arg\max_a Q(S_t, a)$ ▷ Atualiza a política-alvo
        if $A_t \neq \pi(S_t)$ **then**
            break ▷ Interrompe, pois a ação é off-policy
        end if
        $W \leftarrow W \cdot \frac{1}{b(A_t|S_t)}$
    end for
end loop