u évora
ESCOLA DE CIÊNCIAS E TECNOLOGIA
DEPARTAMENTO DE INFORMÁTICA

Sistemas Computacionais de Apoio à Decisão

Miguel Barão

10 de Junho de 2013

Conteúdo

Prefácio

Estas notas foram escritas para a edição de 2012/2013 da disciplina de "Sistemas Computacionais de Apoio à Decisão" no âmbito do programa de Mestrado (eLearning) em Engenharia Informática da Universidade de Évora.

São um texto em desenvolvimento, pelo que incluem certamente muitas gralhas. Agradeço desde já todas as correcções ou sugestões que me sejam enviadas para mjsb@uevora.pt ou partilhadas pela plataforma moodle.

Capítulo 1

Cadeias de Markov Escondidas (HMM)

Neste capítulo iremos estudar cadeias de Markov escondidas (*Hidden Markov Models* ou HMM). As cadeias de Markov são sistemas com estados discretos e com um mecanismo de transição de estado estocástico, descrito por probabilidades de transição. As cadeias de Markov escondidas são semelhantes às anteriores mas, ao contrário destas, o estado não é directamente observado. Por este motivo diz-se que a cadeia de Markov está "escondida".

A maioria dos problemas tratados neste texto são problemas de inferência. Nestes problemas pretende-se *inferir* os estados escondidos da cadeia de Markov a partir das observações. O procedimento de inferência produz uma distribuição de probabilidade que representa o conhecimento adquirido até ao momento, tendo em consideração as observações realizadas. O conhecimento é representado por uma distribuição condicional $p(\text{estado}|\text{observações})$.

Em alguns casos podemos querer *estimar* o estado. O procedimento de *estimação* usa um certo critério para escolher um estado a partir do conhecimento representado numa distribuição de probabilidade como a anterior. Alguns dos critérios possíveis são: escolher o estado com maior probabilidade; o valor esperado; a mediana; *etc.*

1.1 Cadeia de Markov

De um modo geral, um processo de Markov é qualquer processo estocástico que satisfaça a propriedade de Markov. Esta propriedade diz simplesmente que o futuro é condicionalmente independente do passado, dado o presente. Por outras palavras: conhecendo o presente, o passado é irrelevante para fazer predições acerca do futuro.

Muitos processos físicos têm esta propriedade. Por exemplo, para prever a trajectória de uma pedra lançada ao ar, basta conhecer a sua posição e velocidade num certo momento, sendo as posições e velocidades passadas irrelevantes. O mesmo acontece com o jogo do monopólio: a evolução do jogo depende apenas do estado do jogo num dado momento e não do modo como esse estado foi atingido. Tanto no lançamento da pedra como no jogo do monopólio, o futuro não é perfeitamente determinado; existe um elemento de imprevisibilidade que influencia os estados futuros: num caso o vento e turbulência do ar, noutro caso os resultados dos lançamentos futuros dos dados.

Uma cadeia de Markov é um caso particular de um processo de Markov que se caracteriza pelo facto do estado ser discreto: é um elemento de um conjunto finito ou infinito contável (também

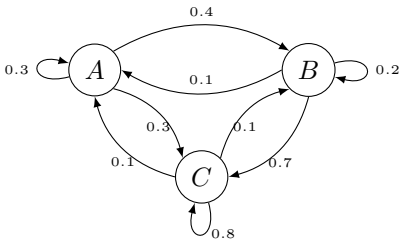


Figura 1.1: Exemplo de uma cadeia de Markov. Os estados estão representados nos nós $\{A, B, C\}$; as probabilidades de transição entre estados estão indicadas nos arcos.

se diz enumerável) de estados possíveis. Neste texto vamos considerar apenas o caso finito.

Uma cadeia de Markov é normalmente caracterizada pelas probabilidades de transição $p(x_{t+1}|x_t)$ entre os estados em instantes de tempo sucessivos. Estas probabilidades podem ser representadas por um grafo como o da figura 1.1, ou representadas numa matriz quadrada

$$P = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{bmatrix}, \tag{1.1}$$

onde $p_{ij} = \Pr(X_{t+1} = j \mid X_t = i)$. Por exemplo, se o estado no instante t é $x_t = 2$, então o estado seguinte pode ser 1, 2 ou 3 com probabilidades

$$\begin{bmatrix} p_{21} & p_{22} & p_{23} \end{bmatrix}, \tag{1.2}$$

obtidas da segunda linha da matriz.

Como os estados futuros dependem apenas do estado presente, o comportamento da cadeia de Markov pode ser caracterizado com memória finita. (Se a propriedade de Markov não fosse satisfeita, seria necessário fazer depender as probabilidades de transição de toda a história passada, o que iria requer memória infinita tanto para guardar a história como para representar as probabilidades de transição $p(x_{t+1}|x_t, x_{t-1}, x_{t-2}, x_{t-3}, \dots)$.)

De um ponto de vista pragmático, mesmo quando um determinado processo não satisfaz a propriedade de Markov, é comum admitir que sim para tornar o problema tratável. Ao efectuar esta aproximação está-se a incorrer num erro (conscientemente e propositadamente). É necessário, nesse caso, verificar se os resultados obtidos com esta aproximação são aceitáveis e, eventualmente, rever todo o procedimento se não o forem.

Um exemplo onde a aproximação anterior é introduzida é o caso da linguagem. Se considerarmos que cada letra do alfabeto é um estado, podemos construir uma cadeia de Markov a partir da frequência com que cada letra surge a seguir a outra na lingua portuguesa. Ao afirmar que a probabilidade de uma letra só depende da letra que ocorreu anteriormente estamos a cometer um erro, pois todas as letras da palavra, e até mesmo das palavras e frases anteriores, deveriam, em princípio, ter sido tidas em conta.

1.2 Caracterização de uma HMM

Na caracterização de uma cadeia de Markov escondida consideram-se dois tipos de variáveis: o estado, representado por x, e a observação representada por y. Assume-se que o estado x

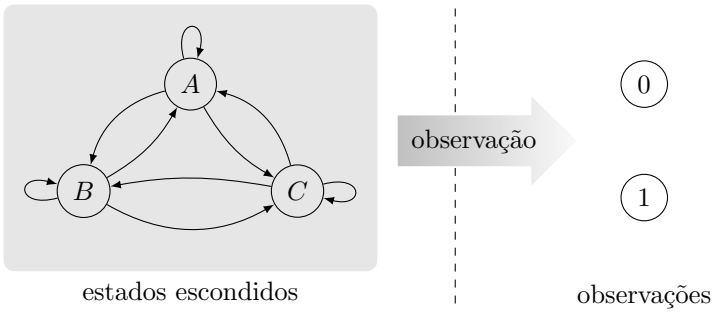


Figura 1.2: Cadeia de Markov escondida (HMM). A parte esquerda é escondida no sentido em que o estado não é directamente observado. No processo de observação são gerados os símbolos 0 ou 1, que dependem probabilisticamente do estado.

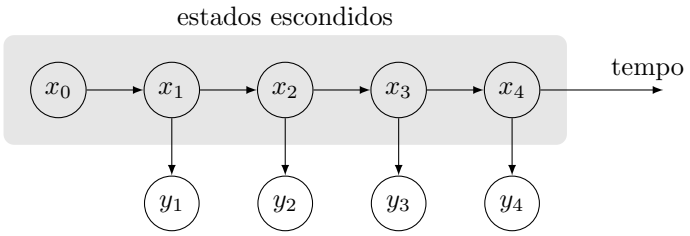


Figura 1.3: Evolução da cadeia de Markov no tempo. As setas representam as dependências entre os estados e observações em instantes de tempo sucessivos. A zona sombreada não está directamente acessível.

pertence a um conjunto finito de estados $\mathcal{X}$, e que a observação y pertence a outro conjunto também finito de possíveis observações $\mathcal{Y}$. Tanto o estado x como as observações y formam um processo estocástico que se desenvolve ao longo do tempo.

Para representar os valores do estado e da observação num determinado instante de tempo t escreve-se respectivamente x_t e y_t. Usa-se a notação $x_{1:t} \triangleq (x_1, \ldots, x_t)$ e $y_{1:t} \triangleq (y_1, \ldots, y_t)$ para representar sequências de estados e sequências de observações, respectivamente.

Para caracterizar o comportamento de uma HMM são necessários dois modelos:

- O *modelo de transição de estado* que contém as probabilidades de transição entre os estados da cadeia de Markov: $p(x_{t+1}|x_t)$.

- O *modelo das observações* que descreve a dependência entre as observações e o estado da cadeia de Markov $p(y_t|x_t)$.

Dependendo do problema considerado, pode ainda ser necessário conhecer a distribuição de probabilidade $p(x_0)$ do estado inicial x_0. As figuras 1.2 e 1.3 mostram um exemplo de uma cadeia de Markov escondida.

1.3 Classes de problemas

De entre os problemas que se colocam sobre HMMs, distinguem-se as seguintes classes de problemas:

Inferência do estado Dada uma cadeia de Markov e uma sequência de observações, como se pode inferir a sequência de estados que melhor as explica?

Estimação do estado Dada uma cadeia de Markov e uma sequência de observações, como se pode estimar a sequência de estados que melhor as explica segundo um certo critério?

Verosimilhança de um modelo Dada uma cadeia de Markov, como se avalia a verosimilhança desse modelo dado um conjunto de observações? (*i.e.*, as observações são bem explicadas pelo modelo?)

Identificação do modelo Dada uma sequência de observações, como se pode estimar os parâmetros da cadeia de Markov?

Os primeiros problemas são relativamente simples, e são tratados neste texto. O último é mais complexo e não é considerado neste curso.

1.4 Inferência do estado de uma HMM

Os problemas de inferência têm como objectivo representar o conhecimento adquirido acerta do estado, a partir de observações feitas ao longo do tempo.

Apresentam-se em seguida três problemas de inferência comuns, que se distinguem pelos tempos (presente, passado, futuro) a que respeitam:

Filtragem Neste problema pretendemos inferir o estado actual com base num conjunto de observações desde o passado até ao presente, isto é, obter a distribuição

$$p(x_t|y_{1:t}). \tag{1.3}$$

Smoothing Neste problema pretendemos inferir um estado passado dadas as observações até ao momento presente. Isto é, obter

$$p(x_k|y_{1:t}), \qquad k < t. \tag{1.4}$$

Predição Neste problema pretende-se predizer o estado futuro, isto é, obter a distribuição

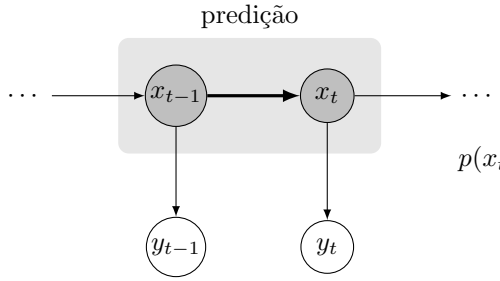
$$p(x_k|y_{1:t}), \qquad k > t. \tag{1.5}$$

Estes problemas são resolvidos nas secções seguintes.

1.5 Filtragem

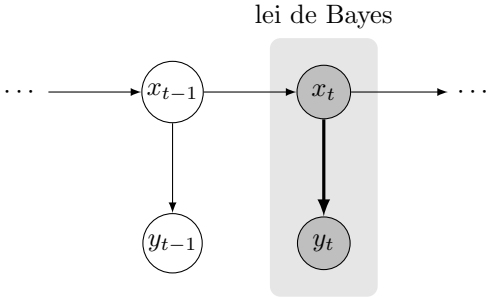
No problema de filtragem pretende-se inferir o estado actual dadas as observações até ao momento, *i.e.*, determinar $p(x_t|y_{1:t})$. Este problema é resolvido em dois passos: predição numa transição da cadeia de Markov e actualização via lei de Bayes.

Predição Usando o conhecimento anterior $p(x_{t-1}|y_{1:t-1})$ e o modelo de transição de estado $p(x_t|x_{t-1})$, efectua-se a predição 1-passo-à-frente $p(x_t|y_{1:t-1})$:



$$p(x_t|y_{1:t-1}) = \sum_{x_{t-1}} p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1})$$

Actualização via lei de Bayes Usando a predição $p(x_t|y_{1:t-1})$ obtida no passo anterior, e usando a nova observação y_t, podemos actualizar o conhecimento acerca do estado actual x_t usando a lei de Bayes:



$$p(x_t|y_{1:t}) = \frac{p(y_t|x_t)p(x_t|y_{1:t-1})}{p(y_t)}$$

Combinando os dois passos numa única fórmula, obtém-se

$$p(x_t|y_{1:t}) = \frac{1}{p(y_t)} p(y_t|x_t) \sum_{x_{t-1}} p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1}). \tag{1.6}$$

A equação (1.6) pode ser vista como uma função recursiva onde $p(x_t|y_{1:t})$ é calculada como função da distribuição $p(x_{t-1}|y_{1:t-1})$ referente ao conhecimento no instante $t-1$, que por sua vez é calculado a partir do conhecimento em $t-2$ e assim sucessivamente até chegar ao conhecimento inicial $p(x_0)$.

Embora a versão recursiva seja interessante do ponto de vista teórico, é computacionalmente ineficiente pois obriga a recalcular toda a recursão de cada vez que se obtém uma nova observação. Em alternativa, pode-se "propagar para a frente" a distribuição $p(x_t|y_{1:t})$ de cada vez que é feita uma nova observação. Nesta abordagem, parte-se do conhecimento inicial $p(x_0)$ e actualiza-se o conhecimento para a distribuição $p(x_1|y_1)$ usando a equação (1.6), após a observação y_1 estar disponível. O procedimento repete-se após cada nova observação $(y_2, y_3, \ldots)$.

Como a propagação da informação se faz para a frente no tempo, o algoritmo é muitas vezes implementado numa função `forward()` que recebe como argumentos

- o conhecimento anterior $p(x_{t-1}|y_{1:t-1})$,
- o modelo de transição de estado $p(x_t|x_{t-1})$,
- o modelo das observações $p(y_t|x_t)$,
- a observação y_t efectuada,

e devolve como resultado o estado de conhecimento actualizado $p(x_t|y_{1:t})$ calculado de acordo com a equação (1.6).

Note que não é necessário representar em memória a distribuição condicional completa, uma vez que se pode substituir a variável y_1 pelo valor efectivamente observado, ficando apenas um vector de probabilidades indexado por x_1.

Exemplo 1. *Uma cadeia de Markov com dois estados A e B transita entre eles com probabilidade* 0.1 *e mantém o estado com probabilidade* 0.9. *No processo de observação, o estado verdadeiro é observado com probabilidade* 0.8. *Considere que o estado inicial é $x_0 = A$. Supondo que foi observada a sequência $ABBB$, o processo de filtragem produz as seguintes distribuições do estado:*

1. *A distribuição inicial do estado é o vector de probabilidades $\begin{bmatrix}1 & 0\end{bmatrix}^\intercal$;*

2. *Foi observado $y_1 = A$. O conhecimento actualizado é*

$$\begin{aligned} p(x_1|y_1 = A) &= \frac{1}{p(y_1 = A)} p(y_1 = A|x_1) \sum_{x_0} p(x_1|x_0)p(x_0) \\ &= \frac{1}{p(y_1 = A)} p(y_1 = A|x_1)p(x_1|x_0 = A). \end{aligned} \tag{1.7}$$

   donde se obtêm as probabilidades $\begin{bmatrix}0.973 & 0.027\end{bmatrix}^\intercal$.

3. *Foi observado $y_2 = B$. O conhecimento actualizado é*

$$p(x_2|y_{1:2} = AB) = \frac{1}{p(y_2 = B)} p(y_2 = B|x_2) \sum_{x_1} p(x_2|x_1)p(x_1|y_1 = A) \tag{1.8}$$

   donde se obtêm as probabilidades $\begin{bmatrix}0.644 & 0.356\end{bmatrix}^\intercal$.

Continuando para as restantes observações, obtém-se a sequência de distribuições

$$\begin{bmatrix} 1.000 & 0.973 & 0.644 & 0.285 & 0.109 \\ 0.000 & 0.027 & 0.356 & 0.715 & 0.891 \end{bmatrix}, \tag{1.9}$$

onde cada coluna contém a distribuição do estado em cada instante de tempo (ver gráfico 1.4).

Repare que após a primeira observação de um B, a filtragem ainda indica que é mais provável o estado x_2 ser o A (como o sensor é mau, dá mais importância ao modelo que prevê que o estado se mantém).

1.6 Smoothing

No problema de *smoothing* pretende-se inferir um estado passado dadas as observações até ao presente, *i.e.*, determinar $p(x_k|y_{1:t})$, $k < t$. Este problema é resolvido por um algoritmo conhecido por *forward-backward*. A ideia por detrás deste algoritmo é em dividir o problema em duas partes: numa faz-se a filtragem usando as observações até ao instante k; na outra usam-se as observações posteriores a k para calcular a verosimilhança do estado no instante k.

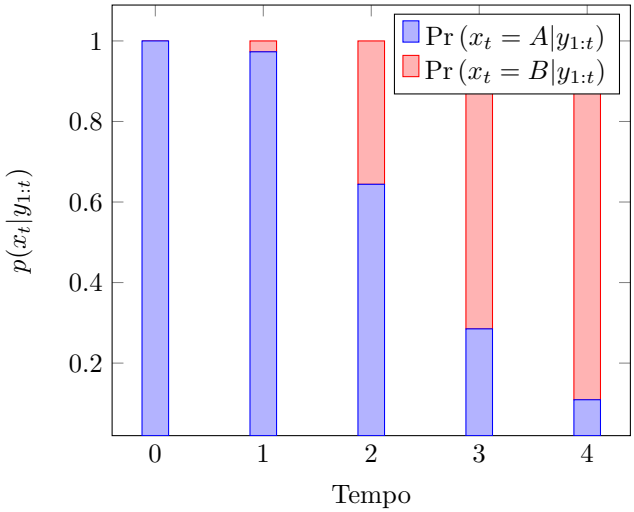


Figura 1.4: Evolução das probabilidades dos estados durante o processo de filtragem quando é observada a sequência $ABBB$.

O algoritmo obtém-se do seguinte modo:

$$\begin{aligned} p(x_k \,|\, y_{1:t}) &= p(x_k \,|\, y_{1:k}, y_{k+1:t}) \\ &= \frac{p(x_k, y_{k+1:t} \,|\, y_{1:k})}{p(y_{k+1:t} \,|\, y_{1:k})} \\ &= \frac{p(x_k \,|\, y_{1:k}) p(y_{k+1:t} \,|\, x_k, y_{1:k})}{p(y_{k+1:t} \,|\, y_{1:k})} \\ &= \frac{p(x_k \,|\, y_{1:k}) p(y_{k+1:t} \,|\, x_k)}{p(y_{k+1:t} \,|\, y_{1:k})} \end{aligned} \tag{1.10}$$

onde a última passagem $p(y_{k+1:t} \,|\, x_k, y_{1:k}) = p(y_{k+1:t} \,|\, x_k)$ é justificada pela independência condicional $y_{k+1:t} \perp\!\!\!\perp y_{1:k} \,|\, x_k$, *i.e.*, as observações posteriores ao instante k são independentes das anteriores dado o estado x_k.

Definindo o factor de normalização $\gamma_k \triangleq p(y_{k+1:t} \,|\, y_{1:k})$, obtemos a equação

$$p(x_k \,|\, y_{1:t}) = \frac{1}{\gamma_k} \underbrace{p(x_k \,|\, y_{1:k})}_{\text{forward}} \underbrace{p(y_{k+1:t} \,|\, x_k)}_{\text{backward}} \tag{1.11}$$

composta pelo produto de dois factores: o *forward* e o *backward*. A componente *forward* é precisamente o problema resolvido na filtragem da secção 1.5. O factor *backward* é calculado do

seguinte modo:

$$\begin{aligned} p(y_{k+1:t} \,|\, x_k) &= \sum_{x_{k+1}} p(y_{k+1:t}, x_{k+1} \,|\, x_k) \\ &= \sum_{x_{k+1}} p(y_{k+1:t} \,|\, x_{k+1}, x_k)\, p(x_{k+1} \,|\, x_k) \\ &= \sum_{x_{k+1}} p(y_{k+1:t} \,|\, x_{k+1})\, p(x_{k+1} \,|\, x_k) && \text{(indep. cond.)} \\ &= \sum_{x_{k+1}} p(y_{k+1}, y_{k+2:t} \,|\, x_{k+1})\, p(x_{k+1} \,|\, x_k) \\ &= \sum_{x_{k+1}} p(y_{k+1} \,|\, x_{k+1})\, p(y_{k+2:t} \,|\, x_{k+1})\, p(x_{k+1} \,|\, x_k). && \text{(indep. cond.)} \end{aligned}$$

Esta fórmula de cálculo é recursiva uma vez que podemos calcular $p(y_{k+1:t} \,|\, x_k)$ a partir de $p(y_{k+2:t} \,|\, x_{k+1})$, e esta por sua vez a partir de $p(y_{k+3:t} \,|\, x_{k+2})$, e por aí adiante até chegarmos ao instante final t.

Tal como acontece no algoritmo *forward*, a implementação recursiva do *backward* não é computacionalmente eficiente pois é necessário recalcular toda a recursão de cada vez que é chamada. Alternativamente, podemos inicializar o algoritmo *backward* no instante final t com

$$p(y_{t+1:t} \,|\, x_t) \triangleq 1 \tag{1.12}$$

e propagar a distribuição $p(y_{k+1:t} \,|\, x_k)$ para trás no tempo a partir de $k = t-1$ usando a fórmula

$$p(y_{k+1:t}|x_k) = \sum_{x_{k+1}} \underbrace{p(y_{k+1}|x_{k+1})}_{\text{modelo obs.}} \underbrace{p(y_{k+2:t}|x_{k+1})}_{\text{recursão}} \underbrace{p(x_{k+1}|x_k)}_{\text{modelo trans.}}. \tag{1.13}$$

A fórmula (1.13) pode ser implementada numa função `backward()` que recebe como argumentos

- a verosimilhança calculada no passo anterior $p(y_{k+2:t}|x_{k+1})$,
- o modelo de transição de estado $p(x_{k+1}|x_k)$,
- o modelo das observações $p(y_{k+1}|x_{k+1})$,
- a observação y_{k+1},

e devolve como resultado $p(y_{k+1:t}|x_k)$ calculado de acordo com a equação (1.13).

Note que não é necessário representar em memória a distribuição condicionada completa, uma vez que se pode substituir $y_{k+1:t}$ pelos valores efectivamente observados, ficando apenas um vector indexado por x_k. Atenção que este vector **não é uma distribuição de probabilidade**!

Finalmente, para calcular a distribuição $p(x_k|y_{1:t})$, com $k < t$, usa-se a fórmula (1.11).

Se o objectivo é obter as distribuições $p(x_k|y_{1:t})$, para todos os $k = 1, \ldots, t$, podemos correr os algoritmos *forward* e *backward* uma única vez, guardando todos os resultados intermédios, e finalmente aplicar a solução (1.11) usando os resultados intermédios guardados.

Exemplo 2. *Uma cadeia de Markov com dois estados A e B transita entre eles com probabilidade* 0.1 *e mantém o estado com probabilidade* 0.9*. No processo de observação, o estado verdadeiro é observado com probabilidade* 0.8*.*

Suponha que foi observada a sequência $y_{1:5} = AABBB$ *e que se pretende inferir o estado* x_3 *no instante de tempo* 3. *Este problema é precisamente o problema de* smoothing *estudado aqui. Isto é, pretende-se calcular*

$$p(x_3|y_{1:5}) = \frac{1}{\gamma_3} p(x_3|y_{1:3}) p(y_{4:5}|x_3).$$

Nota 1: Na resolução que se apresenta em seguida, os dois algorimos forward e backward são aplicados em todo o intervalo de tempo para facilitar a compreensão. Na realidade não vai ser necessário usar toda a informação obtida, uma vez que apenas é necessário aplicar o forward desde $t = 1$ *até* $t = 3$ *e o backward desde* $t = 5$ *até* $t = 3$.

1. *Na condição de falta de informação acerca do estado inicial, assume-se que este tem distribuição uniforme* $p(x_0) = 1/2$.

2. *Aplicando o algoritmo* forward, *obtêm-se as distribuições* $p(x_k|y_{1:k})$, $k = 1, \ldots, 5$, *indicadas nas colunas da matriz seguinte:*

$$\begin{bmatrix} 0.800 & 0.919 & 0.559 & 0.232 & 0.091 \\ 0.200 & 0.081 & 0.441 & 0.768 & 0.909 \end{bmatrix}.$$

   Este algoritmo é precisamente o mesmo usado no problema de filtragem.

3. *Aplicando o algoritmo* backward, *obtêm-se as verosimilhanças* $p(y_{k+1:t}|x_k)$, $k = 1, \ldots, 5$, *indicadas nas colunas da matriz seguinte:*

$$\begin{bmatrix} 0.053 & 0.062 & 0.106 & 0.260 & 1.000 \\ 0.075 & 0.389 & 0.538 & 0.740 & 1.000 \end{bmatrix}.$$

   A última coluna é a coluna inicial (1.12) *do algoritmo. As restantes colunas são calculadas sucessivamente da direita para a esquerda usando a equação* (1.13).

4. *O algoritmo* forward-backward *combina as colunas das duas matrizes anteriores, usando a equação* (1.11), *para se obter*

$$\begin{bmatrix} 0.737 & 0.645 & 0.200 & 0.096 & 0.091 \\ 0.263 & 0.355 & 0.800 & 0.904 & 0.909 \end{bmatrix}.$$

   Cada coluna desta matriz corresponde a um problema de smoothing diferente

$$p(x_k|y_{1:5} = AABBB),$$

   onde o instante k *corresponde à coluna da matriz.*

Nota 2: Pode parecer que percorrendo as várias colunas e escolhendo os estados com maior probabilidade se obtém a melhor a melhor explicação para as observações efectuadas, no entanto este raciocínio está errado! Para obter a melhor explicação é necessário procurar sobre todas as sequências de estados possíveis, isto é sobre a distribuição conjunta $p(x_{1:5}|y_{1:5} = AABBB)$. *Não foi o que se fez no problema de smoothing, pois apenas se considerou cada estado individualmente (distribuição marginal). O raciocínio correcto está explicado na secção 1.8.*

1.7 Predição

De entre os três problemas – filtragem, smoothing e predição – este é o mais simples.

Supondo que foi observada uma sequência $y_{1:t}$, o melhor conhecimento que temos acerca do estado x_t é o que se obtém no problema de filtragem com $p(x_t|y_{1:t})$. Para predizer (ou prever) o estado futuro x_k é necessário determinar

$$p(x_k|y_{1:t}), \quad k > t. \tag{1.14}$$

Vemos imediatamente que o resultado pode ser determinado usando apenas as probabilidades de transição da cadeia de Markov:

$$p(x_k|y_{1:t}) = \sum_{x_{k-1}} p(x_k, x_{k-1}|y_{1:t}) \tag{1.15}$$

$$= \sum_{x_{k-1}} p(x_k|x_{k-1}, y_{1:t}) p(x_{k-1}|y_{1:t}) \tag{1.16}$$

$$= \sum_{x_{k-1}} \underbrace{p(x_k|x_{k-1})}_{\text{prob. trans.}} p(x_{k-1}|y_{1:t}). \tag{1.17}$$

Observa-se que, tal como nos problemas de filtragem e smoothing, obtemos uma fórmula recursiva onde, para calcular $p(x_k|y_{1:t})$, é necessário calcular primeiro $p(x_{k-1}|y_{1:t})$. Esta recursão repete-se até que $k = t$, que é já um problema de filtragem.

Tal como nos outros problemas, a versão recursiva não escala bem computacionalmente. No entanto, pode fazer-se a predição começando com $p(x_t|y_{1:t})$ e depois fazendo predição um passo-à-frente sucessivamente.

Uma característica muito interessante neste algoritmo é o facto de, numa representação matricial, a equação (1.15) corresponder a um produto matricial. Omitindo a parte da filtragem que permite obter a distribuição $p(x_t|y_{1:t})$, isto é, supondo que o estado actual tem distribuição de probabilidade $p(x_t)$, então o estado seguinte tem distribuição de probabilidade

$$p(x_{t+1}) = \sum_{x_t} p(x_{t+1}|x_t) p(x_t). \tag{1.18}$$

Suponhamos que o conjunto de estados é $\{A, B\}$. Então

$$\underbrace{\begin{bmatrix} \Pr(x_{t+1} = A) \\ \Pr(x_{t+1} = B) \end{bmatrix}}_{\boldsymbol{\mu}_{t+1}} = \underbrace{\begin{bmatrix} \Pr(x_{t+1} = A|x_t = A) & \Pr(x_{t+1} = A|x_t = B) \\ \Pr(x_{t+1} = B|x_t = A) & \Pr(x_{t+1} = B|x_t = B) \end{bmatrix}}_{\boldsymbol{P}} \underbrace{\begin{bmatrix} \Pr(x_t = A) \\ \Pr(x_t = B) \end{bmatrix}}_{\boldsymbol{\mu}_t} \tag{1.19}$$

onde $\boldsymbol{\mu}_t$ representa o vector de probabilidades do estado e a matriz $\boldsymbol{P}$ contém as probabilidades de transição da cadeia de Markov.

A predição k-passos-à-frente é então obtida fazendo k predições sucessivas 1-passo-à-frente, que podem ser condensadas numa única operação matricial

$$\boldsymbol{\mu}_{t+k} = \underbrace{\boldsymbol{P}\boldsymbol{P}\cdots\boldsymbol{P}}_{k \text{ vezes}} \boldsymbol{\mu}_t = \boldsymbol{P}^k \boldsymbol{\mu}_t. \tag{1.20}$$

Nota: É usual na literatura definir $\boldsymbol{\mu}$ como uma matriz linha e a matriz de transição $\boldsymbol{P}$ como a transposta da que é indicada em (1.19). Nesse caso escreve-se $\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t \boldsymbol{P}$.

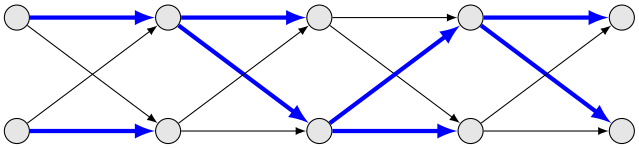


Figura 1.5: Sequência de estados mais verosímil numa cadeia de Markov escondida.

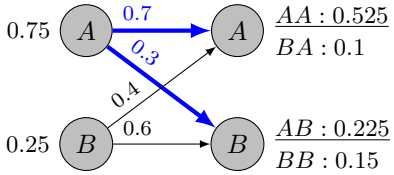


Figura 1.6: Uma transição de uma cadeia de Markov com estados acessíveis.

1.8 Melhor explicação (algoritmo de Viterbi)

Um dos problemas mais comuns em HMMs é o de encontrar a sequência de estados que melhor explica as observações. A melhor explicação é encontrada procurando, de entre todas as sequências de estados, a que tem a maior probabilidade *a posteriori* dadas as observações (critério *maximum a posteriori* ou MAP). Isto é, dadas as observações $y_{1:t}$, determinar

$$x^*_{1:t} = \arg\max_{x_{1:t}} p(x_{1:t}|y_{1:t}), \tag{1.21}$$

Note que a solução deste problema é diferente da obtida resolvendo o problema de *smoothing* separadamente para cada estado x_k e escolhendo o mais provável em cada instante. O que se pretende em (1.21) é encontrar a sequência de estados $x_1, \ldots, x_t$ que maximiza a sua *probabilidade conjunta*, dadas as observações, enquanto que no problema de *smoothing* se calculam as *distribuições marginais* dos estados dadas as observações.

O problema (1.21) tem um algoritmo simples baseado em programação dinâmica chamado *algoritmo de Viterbi*, em homenagem ao seu autor [**?**]. O problema é colocado como a procura de um caminho num grafo como o da figura 1.5.

O algoritmo tem complexidade computacional e de memória linear com o horizonte t considerado, e é usado em muitas aplicações na área das telecomunicações como por exemplo em televisão digital, comunicações por satélite, discos rígidos e reconhecimento de fala.

Antes de resolver o problema completo sobre uma HMM, vamos começar por considerar uma cadeia de Markov sem estados escondidos e apenas uma transição, como mostra a figura 1.6.

Suponhamos que $p(x_0)$ é conhecido, assim como as probabilidades de transição $p(x_1|x_0)$, ambas indicadas na figura 1.6. Equipados com esta informação, pretendemos determinar a sequência de estados com maior probabilidade de ocorrer. Respondemos a esta questão construindo a probabilidade conjunta $p(x_0, x_1)$ e determinando qual a sequência (x_0, x_1) que a maximiza, isto é,

$$\max_{x_0,x_1} p(x_0, x_1) = \max_{x_0,x_1} p(x_1|x_0)p(x_0). \tag{1.22}$$

No exemplo da figura 1.6 representam-se as várias transições de estado possíveis e as respectivas probabilidades associadas. À direita estão representadas as probabilidades conjuntas das sequências AA, AB, BA e BB.

Considerando o estado terminal $x_1 = A$, a sequência mais provável para o atingir é a sequência AA, pois tem a probabilidade mais elevada de entre as duas sequências que atingem $x_1 = A$. Da mesma forma, para o estado terminal $x_1 = B$, a sequência mais provável é AB. As sequências mais prováveis que terminam respectivamente em A e B estão sublinhadas na figura e foram obtidas com

$$\max_{x_0} p(x_0, x_1) = \max_{x_0} p(x_1|x_0)p(x_0). \tag{1.23}$$

De entre todas as quatro sequências, a mais provável é a AA e é obtida com

$$\max_{x_0,x_1} p(x_0, x_1) = \max_{x_1} \Big(\max_{x_0} p(x_1|x_0)p(x_0) \Big). \tag{1.24}$$

Passemos agora para a aplicação do mesmo conceito em HMMs. A grande diferença relativamente ao problema anterior é a dependência de observações por via da lei de Bayes.

Considerando apenas uma transição, a sequência com maior probabilidade de atingir um certo estado x_1 é

$$\begin{aligned} \max_{x_0} p(x_0, x_1|y_1) &= \max_{x_0} \Big(\frac{1}{\gamma_1} p(y_1|x_0, x_1)p(x_1|x_0)p(x_0) \Big) \\ &= \max_{x_0} \Big(\frac{1}{\gamma_1} p(y_1|x_1)p(x_1|x_0)p(x_0) \Big) \\ &= \frac{1}{\gamma_1} p(y_1|x_1) \max_{x_0} \Big(p(x_1|x_0)p(x_0) \Big), \end{aligned} \tag{1.25}$$

em que γ_1 é um factor de normalização. Note que a expressão $\max_{x_0} p(x_0, x_1|y_1)$, após a observação ser feita, depende apenas de x_1, podendo ser representada num vector de probabilidades.

A sequência mais provável pode ser obtida calculando, em seguida, o máximo do vector de probabilidades relativamente a x_1. Mas não vamos fazer isso já! Iremos primeiro estender o problema para o caso em que há duas transições de estado e duas observações.

A sequência mais provável que termina num certo estado x_2 é obtida maximizando a distribuição conjunta

$$\max_{x_0,x_1} p(x_0, x_1, x_2|y_{1:2}) = \frac{1}{\gamma_2} p(y_2|x_2) \max_{x_1} \Big(p(x_2|x_1) \max_{x_0} p(x_0, x_1|y_1) \Big), \tag{1.26}$$

em que o factor da direita é precisamente o que foi determinado em (1.25). Continuando pelo mesmo raciocínio chegamos à fórmula geral recursiva

$$\max_{x_{0:t-1}} p(x_{0:t}|y_{1:t}) = \frac{1}{\gamma_t} p(y_t|x_t) \max_{x_{t-1}} \Big(p(x_t|x_{t-1}) \max_{x_{0:t-2}} p(x_{1:t-1}|y_{1:t-1}) \Big), \tag{1.27}$$

em que o cálculo do máximo $p(x_{0:t}|y_{1:t})$ é feito recursivamente a partir do máximo de $p(x_{1:t-1}|y_{1:t-1})$, que é precisamente a solução proposta pela programação dinâmica.

Para obter a explicação mais verosímil, começamos por calcular progressivamente

$$\max_{x_0} p(x_{0:1}|y_1), \quad \max_{x_{0:1}} p(x_{0:2}|y_{1:2}), \quad \ldots, \quad \max_{x_{0:t-1}} p(x_{0:t}|y_{1:t}). \tag{1.28}$$

Após o último passo, calculamos máximo relativamente a x_t, uma vez que não foi incluído em (1.28). Nesse momento ficamos com a maior probabilidade $p(x_{0:t}|y_{1:t})$ sobre todas as possíveis sequências $x_{0:t}$. Para obtermos a sequência correspondente a este máximo, é necessário andar para trás no tempo a partir do último estado x_t e recuperando todos os x_k intermédios que foram obtidos em cada maximização. Para isto ser possível, é necessário que durante a fase *"forward"* do algoritmo, se guardem, não só os valores dos máximos $\max_{x_{0:k-1}} p(x_{0:k}|y_{1:k})$, mas também os estados que dão origem a esses máximos.

O procedimento que acabámos de descrever é precisamente o *algoritmo de Viterbi*.

Exemplo 3. *Uma cadeia de Markov com dois estados A e B transita entre eles com probabilidade* 0.1 *e mantém o estado com probabilidade* 0.9*. No processo de observação, o estado verdadeiro é observado com probabilidade* 0.8*. (mesmo problema das secções anteriores)*

Suponha que foi observada a sequência $y_{1:5} = AABBB$ e que se pretende encontrar a sequência de estados $x_{1:5}$ que melhor explica as observações.

1. *Na condição de falta de informação acerca do estado inicial x_0, assume-se que este tem distribuição inicial uniforme $p(x_0) = 1/2$.*

2. *Vamos calcular agora*

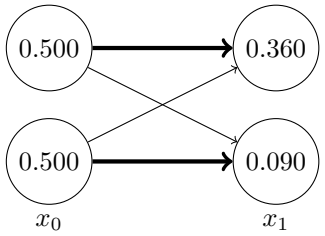
$$\max_{x_0} p(x_0, x_1|y_1) = \frac{1}{\gamma_1} p(y_1|x_1) \max_{x_0} \Big(p(x_1|x_0)p(x_0) \Big). \tag{1.29}$$

Representando as distribuições matricialmente tal que x_1 indexa as linhas e x_0 indexa a coluna da matriz, obtém-se

$$\frac{1}{\gamma_1} \begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix} \circ \max_{x_0} \begin{bmatrix} 0.9 \times \frac{1}{2} & 0.1 \times \frac{1}{2} \\ 0.1 \times \frac{1}{2} & 0.9 \times \frac{1}{2} \end{bmatrix} = \frac{1}{\gamma_1} \begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix} \circ \begin{bmatrix} 0.45 \\ 0.45 \end{bmatrix} = \frac{1}{\gamma_1} \begin{bmatrix} 0.36 \\ 0.09 \end{bmatrix}, \tag{1.30}$$

onde o operador $\circ$ representa o produto elemento-a-elemento (chamado produto de Hadamard). Repare também que não foi necessário determinar o factor de normalização γ_1. Para simplificar esta resolução, omitimos as normalizações pois estas não afectam o resultado da maximização.

Este primeiro passo pode ser representado no grafo



onde as setas a negrito correspondem aos x_0 que foram argumento da maximização (o x_0 que maximizou cada uma das linhas da matriz na equação (1.30)). Após substituir a observação y_1 na expressão $\max_{x_0} p(x_0, x_1|y_1)$, esta é uma função apenas de x_1. O valor calculado é o que está nos dois nós da direita do grafo anterior (sem normalização).

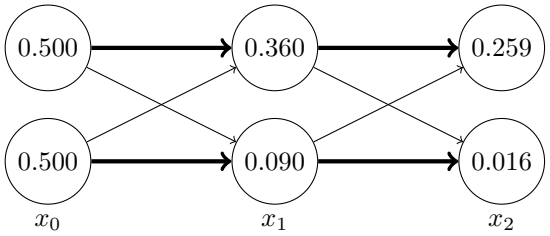
3. *Vamos calcular agora o segundo passo, com $y_2 = A$.*

$$\max_{x_{0:1}} p(x_0, x_1, x_2|y_{1:2}) = \frac{1}{\gamma_2} p(y_2|x_2) \max_{x_1} \Big(p(x_2|x_1) \underbrace{\max_{x_0} p(x_0, x_1|y_1)}_{\text{do passo anterior}} \Big). \tag{1.31}$$

Usando novamente a notação matricial obtém-se

$$\frac{1}{\gamma_2} \begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix} \circ \max_{x_1} \begin{bmatrix} 0.9 \times 0.36 & 0.1 \times 0.09 \\ 0.1 \times 0.36 & 0.9 \times 0.09 \end{bmatrix} = \frac{1}{\gamma_2} \begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix} \circ \max_{x_1} \begin{bmatrix} \mathbf{0.324} & 0.009 \\ 0.036 & \mathbf{0.081} \end{bmatrix} =$$
$$= \frac{1}{\gamma_2} \begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix} \circ \begin{bmatrix} 0.324 \\ 0.081 \end{bmatrix} \approx \frac{1}{\gamma_2} \begin{bmatrix} 0.259 \\ 0.016 \end{bmatrix}. \tag{1.32}$$

Adicionando este passo ao grafo obtém-se



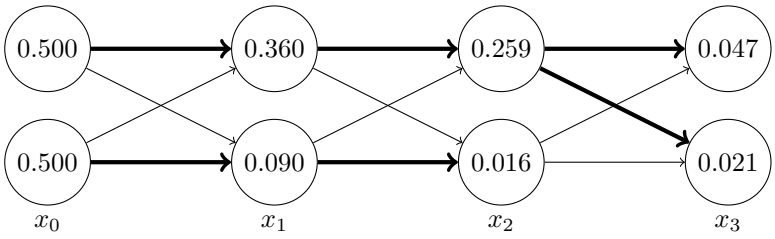
4. *Neste passo foi observado $y_3 = B$. Calcula-se então*

$$\max_{x_{0:2}} p(x_{0:3}|y_{1:3}) = \frac{1}{\gamma_3} p(y_3|x_3) \max_{x_2} \Big(p(x_3|x_2) \underbrace{\max_{x_{0:1}} p(x_{0:2}|y_{1:2})}_{\textit{do passo anterior}} \Big). \tag{1.33}$$

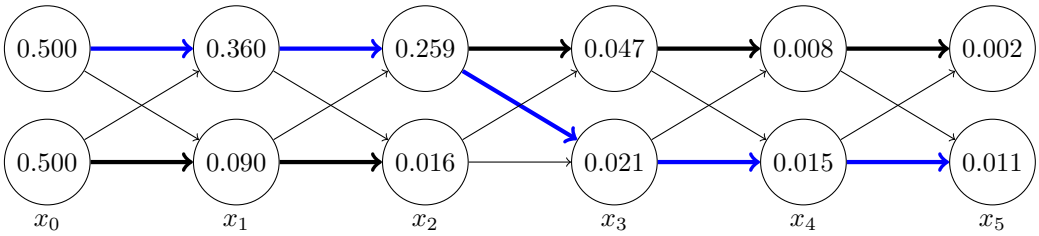
Ou em notação matricial,

$$\frac{1}{\gamma_3} \begin{bmatrix} 0.2 \\ 0.8 \end{bmatrix} \circ \max_{x_2} \begin{bmatrix} 0.9 \times 0.259 & 0.1 \times 0.016 \\ 0.1 \times 0.259 & 0.9 \times 0.016 \end{bmatrix} \approx \frac{1}{\gamma_3} \begin{bmatrix} 0.2 \\ 0.8 \end{bmatrix} \circ \max_{x_1} \begin{bmatrix} \mathbf{0.233} & 0.002 \\ \mathbf{0.026} & 0.014 \end{bmatrix} =$$
$$= \frac{1}{\gamma_3} \begin{bmatrix} 0.2 \\ 0.8 \end{bmatrix} \circ \begin{bmatrix} 0.233 \\ 0.026 \end{bmatrix} \approx \frac{1}{\gamma_3} \begin{bmatrix} 0.047 \\ 0.021 \end{bmatrix}. \tag{1.34}$$

Adicionando este passo ao grafo obtém-se



5. *O procedimento anterior é prosseguindo até ao instante final correspondente a x_5. No final obtém-se um grafo como o seguinte:*



Recupera-se a sequência de estados que levou ao máximo a posteriori escolhendo o máximo no último instante, neste caso 0.011*, e depois andando para trás no tempo (backtracking) até ao instante inicial. A melhor explicação é a que começa com $x_0 = A$ e depois segue a sequência $AABBB$.*

Repare que se tivessemos apenas usado as observações até y_3 a melhor explicação teria sido diferente: $x_0 = A$ seguido de AAA, isto é, a observação $y_3 = B$ seria considerada

como um erro de observação uma vez que a probabilidade do estado se manter é elevada. No entanto, as observações posteriores a y_3 fizeram "mudar de opinião" e considerar que ocorreu de facto uma transição no estado nessa altura.

A resolução apresentada acima sofre de problemas numéricos (underflow) quando o horizonte temporal é grande. Este problema pode ser resolvido fazendo as operações em termos do logaritmo das probabilidades. Esta e outras técnicas são descritas na secção seguinte.

1.9 Implementação/problemas numéricos

A implementação prática do algoritmo de Viterbi e dos algoritmos de filtragem e smoothing resulta frequentemente em problemas numéricos de *underflow/overflow*. Isto deve-se ao facto de existirem produtos sucessivos de probabilidades, todas elas no intervalo $[0, 1]$, levando a que o resultado se aproxime de zero à medida que o horizonte temporal t aumenta. Quando a resolução numérica deixa de ser suficiente, ocorre o *underflow*. Se o *underflow* ocorre num denominador, então obtemos um *overflow* ou uma divisão por zero.

Para mitigar o *underflow*, é comum usar alguns truques:

- Recorrer a normalizações, se isso não afectar o resultado do algoritmo.
- Efectuar cálculos em termos dos logaritmos das probabilidades de modo a converter os produtos em somas. Isto é, em vez de fazer o cálculo usando as probabilidades directamente

$$p_1\, p_2\, \cdots\, p_t,$$

usamos o logaritmo da probabilidade $l_i \triangleq \log p_i$ de modo a obter

$$\log(p_1\, p_2\, \cdots\, p_t) = l_1 + l_2 + \cdots + l_t.$$

- Usar um truque conhecido por *log-sum-exp*. Sempre que existe uma expressão do tipo

$$a = \log \sum_t \exp(b_t),$$

onde os b_t são números muito pequenos ($\to -\infty$), a exponencial resulta em *underflow*. Para o evitar, podemos definir uma constante $B \triangleq \max_t b_t$, e usar as seguintes igualdades

$$\begin{aligned} a &= \log\left(\sum_t e^{b_t}\right) \\ &= \log\left(\sum_t e^{b_t - B + B}\right) \\ &= \log\left(\sum_t e^{b_t - B} e^{B}\right) \\ &= \log\left(e^{B} \sum_t e^{b_t - B}\right) \\ &= B + \log\left(\sum_t e^{b_t - B}\right). \end{aligned} \tag{1.35}$$

A ideia é mover os valores b_t para perto de zero com $b_t - B$, onde a exponencial é bem comportada numericamente. Esta modificação no argumento da exponencial é compensada no final com a adição de B.

Estes truques trocam mau comportamento numérico por maior complexidade computacional. Vejamos um exemplo simples

Exemplo 4. *Sejam* $b_1 = -1001$ *e* $b_2 = -1002$. *Calculando directamente obtém-se (em Python):*

```
>>> b1 = -1001
>>> b2 = -1002
>>> log(exp(b1) + exp(b2))
-inf
```

Ambas as exponenciais resultaram em underflow*, seguido de um* $\log(0)$ *que gerou o símbolo* `-inf`*. Usando o truque* log-sum-exp *obtém-se*

```
>>> B = max(b1, b2)
>>> B + log(exp(b1 - B) + exp(b2 - B))
-1001.4586751453871
```

que é o resultado correcto.

1.10 Exercícios

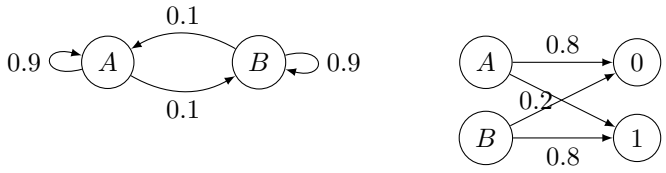
Exercício 1. *Considere uma cadeia de Markov escondida com espaço de estados* $\mathcal{S} = \{A, B\}$ *e onde as probabilidades de transição para o mesmo estado são* 0.8. *Esta cadeia de Markov é observada por uma variável* y_t*, dependente do estado* x_t*, com probabilidades condicionadas*

$y_t \backslash x_t$	A	B
0	*0.9*	*0.5*
1	*0.1*	*0.5*

Supondo o estado inicial x_0 *desconhecido, e tendo sido observada a sequência* 0011*, actualize a distribuição de probabilidade* $p(x_t)$ *que representa o conhecimento (ou crença) sobre o estado em cada instante de tempo.*

Supondo que o sensor avaria depois da quarta observação, como se actualizam as probabilidades daí para a frente? Nestas condições, $p(x_t)$ *converge para qual distribuição quando* $t \to \infty$*?*

Exercício 2. *Considere uma cadeia de Markov escondida cujo modelo de transição e modelo de observações é dado na figura abaixo.*



Sabendo que foi feita a sequência de observações $(0, 1, 1, 1)$*, calcule a probabilidade do estado* $p(x_k|y_{1:4})$*, para* $k = 1, \ldots, 4$.

Exercício 3. *Considerando a cadeia de Markov escondida do exercício 2, calcule a sequência de estados mais provável para a observação* $(0, 1, 1, 1)$.

Capítulo 2

Processos de Decisão de Markov (MDP)

2.1 Introdução

Os Processos de Decisão de Markov (*Markov Decision Processes* ou MDP) são uma abordagem matemática para representar tomadas de decisão em problemas envolvendo incerteza.

Tipicamente consideramos um processo cuja evolução no tempo depende das acções tomadas por um agente de decisão. São tomadas decisões em sequência em instantes de tempo discretos e são recebidas recompensas (ou penalizações). O objectivo é a maximização das recompensas recebidas (ou minimização das penalizações/custos) ao longo de um horizonte de tempo.

Os processos de decisão de Markov são uma extensão das cadeias de Markov pela adição de acções externas. Estas acções permitem influenciar as transições entre os estados do sistema e deste modo atingir objectivos predeterminados.

Existem inúmeras variantes de MDPs: tempo contínuo ou discreto, estados contínuos ou discretos, acções determinísticas ou estocásticas e vários tipos de funções de recompensa. Dá-se em seguida um exemplo de um processo de decisão de Markov.

Exemplo 5. *Uma empresa com uma frota de veículos decide mensalmente quais os veículos que devem ser substituídos por veículos novos. A decisão de substituir ou não um veículo depende do estado desse mesmo veículo. O estado do veículo deve reflectir a probabilidade de avaria e custos de manutenção associados. O objectivo é minimizar os custos sendo necessário decidir em que momento da vida de um veículo este deve ser substituído.*

Neste problema, podemos considerar que o estado de um veículo é representado por uma variável contínua que representa o seu desgaste. O tempo é discreto e coincide com os momentos de tomada de decisão. As acções são também discretas: substituir ou não substituir um veículo.

Neste texto vamos considerar exclusivamente MDPs com um número finito de estados e de acções.

2.2 Formulação do modelo

A caracterização de um MDP requer a especificação dos seguintes itens:

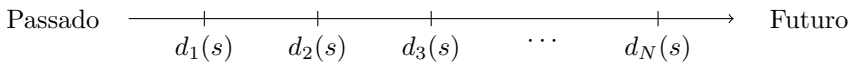


Figura 2.1: Em cada instante de decisão é tomada uma acção a que depende da regra de decisão $d_t(x_t)$ em efeito nesse instante.

Instantes de decisão (ou épocas) são os momentos em que são escolhidas e tomadas acções. Os instantes de decisão são indexados por números inteiros que introduzem a ordem temporal pela qual as decisões são tomadas. Estes números não necessitam de representar unidades de tempo físicas (segundos, horas, dias, *etc*) e não necessitam de corresponder a intervalos de tempo constantes.

O conjunto dos instantes de decisão pode ser:

- *finito* $\mathcal{T} = \{1, 2, \ldots, N\}$;
- *infinito enumerável* $\mathcal{T} = \{1, 2, \ldots\}$.

Alguns exemplos de instantes de decisão são:

- Numa sequência de consultas médicas, o médico decide qual a acção seguinte a tomar para curar um paciente. Os instantes de decisão são as consultas.
- Um condutor em viagem tem como instantes de decisão os momentos em que se encontra nalgum cruzamento e precisa de decidir o caminho a seguir em seguida.
- Um gestor de uma loja decide semanalmente quantos itens deve adquirir para repor o stock num nível adequado.

Estados são as várias configurações em que um sistema se pode encontrar nos instantes de decisão. Em cada instante de decisão o sistema encontra-se num estado bem definido embora este possa não ser completamente conhecido.

- O conjunto dos estados é representado por $\mathcal{X}$.
- Um estado particular é representado por x, com $x \in \mathcal{X}$.
- Num dado instante de decisão t o sistema encontra-se num certo estado x_t.

Alguns exemplos de estados são:

- Um paciente pode estar num dos estados de $\mathcal{X} = \{\text{Doente}, \text{Saudável}\}$.
- Um condutor necessita de guiar em função do local onde se encontra. Os estados são os vários cruzamentos existentes no mapa e nos quais é necessário tomar uma acção (sobre qual a estrada a seguir).
- A quantidade de itens que um armazenista tem em stock num dado momento é um estado. Neste caso, o conjunto dos estados possíveis é $\mathcal{X} = \{0, 1, 2, 3, \ldots, N\}$ onde N é o número de itens que cabem no armazém.

Acções são o mecanismo pelo qual um sistema por ser actuado nos instantes de decisão.

Em cada instante de decisão, um agente toma uma acção $a \in \mathcal{A}_x$ de um conjunto de acções disponíveis $\mathcal{A}_x$. O conjunto das acções disponíveis pode depender do estado x em que o sistema se encontra nesse momento. Caso o conjunto de acções disponíveis seja o mesmo para todos os estados, o conjunto representa-se simplesmente por $\mathcal{A}$.

Como exemplos de acções temos:

- O médico decide qual o antibiótico a tomar: A, B, ou nenhum.
- O condutor decide seguir em frente ou virar à direita. Aqui o conjunto das acções depende do estado (depende dos caminhos possíveis em cada cruzamento).
- O gestor da loja decide comprar A itens para repor o stock. As acções são o número de itens a adquirir.

Probabilidades de transição caracterizam o comportamento do sistema ao longo do tempo. São semelhantes às probabilidades de transição das cadeias de Markov com a diferença de dependerem também das acções.

Formalmente, as probabilidades de transição são probabilidades condicionadas

$$p_t(x_{t+1}|x_t, a_t), \tag{2.1}$$

onde a_t é a acção tomada e x_t é o estado, ambos no instante t.

Recompensa é o valor recebido imediatamente como prémio num certo instante de decisão t. Podemos ter vários tipos de recompensas: a recompensa ser uma função apenas do estado

$$r_t(x_t), \tag{2.2}$$

ou depender também da acção nesse instante

$$r_t(x_t, a_t), \tag{2.3}$$

ou do estado seguinte

$$r_t(x_t, a_t, x_{t+1}). \tag{2.4}$$

Em certos problemas pode fazer mais sentido considerar custos em vez de recompensas. Formalmente, os MDPs podem lidar com qualquer um dos pontos de vista. Para transformar um custo numa recompensa basta trocar o sinal, *e.g.*, um custo de 3 é uma recompensa de -3.

2.3 Terminologia

Para além da caracterização de um MDP, como efectuado na secção anterior, é necessário introduzir um conjunto de termos adicionais.

História é a colecção $h_t = (x_1, a_1, \ldots, x_{t-1}, a_{t-1}, x_t)$ de acções e estados passados. O espaço das histórias até ao instante t é representado por $\mathcal{H}_t$.

Regra de decisão determinística é uma regra que associa a cada história uma acção determinística $d_t : \mathcal{H}_t \to \mathcal{A}_{x_t}$.

Regra de decisão estocástica é uma regra que associa a cada história uma distribuição de probabilidade no conjunto das acções $d_t : \mathcal{H}_t \to \mathcal{P}(\mathcal{A}_{x_t})$. Neste caso, uma acção é gerada aleatoriamente de acordo com a distribuição de probabilidade especificada pela regra de decisão.

Regra de decisão Markoviana é uma regra de decisão que depende apenas do estado actual do sistema e não da história passada. Ou seja, são regras de decisão $d_t : \mathcal{X} \to \mathcal{A}$ ou $d_t : \mathcal{X} \to \mathcal{P}(\mathcal{A})$, consoante a regra é determinística ou estocástica, respectivamente.

As regras de decisão podem ser classificadas nas duas características relativas ao determinismo e à historia:

	Determinística	Estocástica
Markoviana	X	
Dependente da história		

Neste texto considera-se apenas a versão assinalada com X.

Política/Estratégia é a especificação das regras de decisão a usar em cada instante de tempo. Não são as acções propriamente ditas que são especificadas, mas sim o mecanismo de selecção das acções em função do estado. A política é representada pela letra π:

$$\pi = \big(d_1(x), d_2(x), \ldots, d_N(x)\big). \tag{2.5}$$

Política óptima é a política π^* que optimiza uma certa função. Normalmente, a função a optimizar é uma soma de recompensas que se espera virem a ser obtidas no futuro ao seguir uma certa política.

2.4 Horizonte temporal finito

As políticas óptimas considerando horizontes de tempo finitos são geralmente políticas não estacionárias. Isto é, as regras de decisão variam ao longo do tempo:

$$\pi = \big(d_1(x), d_2(x), d_3(x), \ldots, d_{N-1}(x), d_N(x)\big). \tag{2.6}$$

Este facto é facilmente compreensível uma vez que que no instante terminal N não são consideradas recompensas futuras e, portanto, as acções têm apenas em conta a recompensa imediata. Em instantes intermédios são consideradas, não só as recompensas imediatas, mas também as recompensas futuras esperadas até ao instante terminal. Assim, as decisões são moldadas de forma diferente ao longo do horizonte de tempo.

A política óptima pode ser obtida por programação dinâmica como se mostra no exemplo seguinte.

Exemplo 6. *Uma cadeia de Markov com dois estados transita entre eles com probabilidades que dependem de uma acção externa $a \in \{0, 1\}$. Se $a = 0$ o estado mantém-se com probabilidade 0.9, enquanto que se $a = 1$ o estado mantém-se ou comuta com probabilidade 0.5. Suponhamos que as recompensas imediatas são dadas por*

$$r(x) = \begin{cases} 1 & \text{se } x = 0, \\ 5 & \text{se } x = 1. \end{cases} \tag{2.7}$$

Vamos considerar duas situações: Na primeira pretende-se optimizar a recompensa imediata, enquanto na segunda optimiza-se a recompensa total esperada em dois instantes de tempo (com duas acções em sequência). Em ambos os casos começamos no estado inicial $x_0 = 0$.

- *Começamos por analizar o problema com apenas uma transição. A recompensa esperada caso se escolha a acção a_0 é*

$$\mathsf{E}\big[r_1(x_1)\big] = \sum_{x_1} p(x_1|x_0, a_0) r_1(x_1), \tag{2.8}$$

ou seja, se optarmos por $a_0 = 0$, então a recompensa esperada é $0.9 \times 1 + 0.1 \times 5 = 1.4$, enquanto que se optarmos por $a_0 = 1$, então a recompensa esperada é $0.5 \times 1 + 0.5 \times 5 = 3$. A acção óptima é obtida com

$$a_0^* = \arg\max_{a_0} \sum_{x_1} p(x_1|x_0, a_0) r_1(x_1) = 1. \tag{2.9}$$

Esta solução é válida apenas para horizontes de tamanho $N = 1$. Para horizontes maiores é necessário ter em consideração o futuro como se mostra em seguida.

- *Considerando um horizonte de duas transições, obtemos uma recompensa total esperada de*

$$\begin{aligned} \mathsf{E}\big[r_1(x_1) + r_2(x_2)\big] &= \sum_{x_1, x_2} p(x_1, x_2|x_0, a_0, a_1)\big(r_1(x_1) + r_2(x_2)\big) && (2.10)\\ &= \sum_{x_1, x_2} p(x_2|x_1, a_1) p(x_1|x_0, a_0)\big(r_1(x_1) + r_2(x_2)\big) && (2.11)\\ &= \sum_{x_1} p(x_1|x_0, a_0) \left(r_1(x_1) + \sum_{x_2} p(x_2|x_1, a_1) r_2(x_2) \right). && (2.12) \end{aligned}$$

Para obter a acção a_0 óptima é necessário primeiro conhecer o valor da expressão entre parentesis de (2.12). Como essa expressão depende da acção seguinte a_1, é necessário primeiro determinar a acção óptima a_1 e só depois a acção a_0. Formalmente,

$$\max_{a_0, a_1} \mathsf{E}\big[r_1(x_1) + r_2(x_2)\big] = \max_{a_0} \sum_{x_1} p(x_1|x_0, a_0) \underbrace{\left(r_1(x_1) + \max_{a_1} \sum_{x_2} p(x_2|x_1, a_1) r_2(x_2) \right)}_{\triangleq U_1(x_1)}. \tag{2.13}$$

A expressão $U_1(x_1)$ na equação (2.13) do exemplo anterior é chamada a *utilidade do estado*. A utilidade do estado contabiliza as recompensas que esperamos vir a obter a partir desse estado se seguirmos a política óptima daí para a frente. Usando esta noção de utilidade, podemos generalizar para horizontes temporais de qualquer tamanho finito N. Em qualquer instante de tempo t, a acção óptima a_t^* é a que maximiza a utilidade esperada

$$a_t^* = \arg\max_{a_t} \sum_{x_{t+1}} p(x_{t+1}|x_t, a_t) U_{t+1}(x_{t+1}). \tag{2.14}$$

A partir do momento em que a acção a_t^* está determinada, a utilidade do estado x_t é obtida somando a recompensa imediata $r_t(x_t)$ com a utilidade esperada (*i.e.*, as recompensas que se esperam vir a obter no futuro):

$$U_t(x_t) = r_t(x_t) + \max_{a_t} \sum_{x_{t+1}} p(x_{t+1}|x_t, a_t) U_{t+1}(x_{t+1}). \tag{2.15}$$

Conclui-se que a utilidade se propaga para trás no tempo e que as acções óptimas são também obtidas andando para trás. Como o problema é formulado num intervalo de tempo finito, a propagação é inicializada no instante terminal com $U_N(x_N) = r_N(x_N)$.

2.5 Horizonte temporal infinito

No caso de um horizonte temporal infinito a política óptima é estacionária:

$$\pi = \big(d(x), d(x), d(x), d(x), \dots\big). \tag{2.16}$$

Neste caso, não existe razão para considerar regras de decisão diferentes em instantes de tempo diferentes, pelo que as regras de decisão mantêm-se constantes. Assim, com um horizonte infinito, obter a política óptima consiste em encontrar apenas uma regra de decisão $d(x)$, que é então aplicada em todos os instantes de decisão.

Para encontrar a regra decisão $d(x)$ seguimos novamente o princípio da *máxima utilidade esperada* (relembrando, a utilidade $U(x)$ de um estado x indica quanto esperamos vir a ganhar no futuro por estarmos agora nesse estado). Vimos anteriormente que, com horizonte temporal finito, a optimização pode ser feita começando no instante terminal e "andando para trás" até ao presente, alternando as equações (2.14) e (2.15). Infelizmente, no caso de um horizonte infinito, não é possível inicializar o algoritmo no instante terminal (é infinito), pelo que este método não é aplicável. Em alternativa, podemos tentar calcular a utilidade dos estados $U(x)$, que neste caso se assume constante no tempo. Para obtermos a utilidade, temos de ter em consideração a recompensa imediata mais as recompensas que esperamos vir a obter no futuro. Aqui surge um problema técnico: a soma das recompensas num horizonte de tempo infinito pode divergir. Para evitar esta situação, as recompensas futuras são "descontadas" de um factor $0 \leq \gamma < 1$.

Nesta reformulação, a utilidade de uma sequência temporal de estados $(x_0, x_1, x_2, \dots)$ é definida como a recompensa imediata $r(x_0)$ mais a utilidade futura descontada:

$$U(x_0, x_1, x_2, \dots) = r(x_0) + \gamma U(x_1, x_2, x_3, \dots), \tag{2.17}$$

onde, se expandirmos a utilidade do lado direito, obtemos que

$$U(x_0, x_1, x_2, \dots) = r(x_0) + \gamma r(x_1) + \gamma^2 r(x_2) + \gamma^3 r(x_3) + \cdots . \tag{2.18}$$

O facto de o desconto γ ser inferior à unidade faz com que as recompensas num futuro distante sejam menos consideradas, mas tem a vantagem que garantir convergência da função de utilidade. No caso extremo de $\gamma = 0$, apenas se dá importância ao presente (viver o momento), é um problema simples mas de pouco interesse pois o horizonte temporal é irrelevante para a decisão.

Vamos agora tentar resolver o problema de obter a utilidade dos estados de modo a poder optimizar-se esta função e obter as acções óptimas.

Considerando um sistema com n estados $\mathcal{X} = \{x^1, \dots, x^n\}$, a utilidade de cada um é calculada usando a equação de Bellman (2.15). Como para horizontes infinitos a utilidade é constante, obtemos um sistema de n equações a n incógnitas, onde as incógnitas são precisamente a utilidade de cada um dos n estados:

$$\begin{aligned} U(x^1) &= \max_a \left[r(x^1) + \gamma \sum_{x' \in \mathcal{X}} p(x' \mid x^1, a) U(x') \right], \\ &\vdots \\ U(x^n) &= \max_a \left[r(x^n) + \gamma \sum_{x' \in \mathcal{X}} p(x' \mid x^n, a) U(x') \right]. \end{aligned} \tag{2.19}$$

Infelizmente estas equações são não-lineares, devido ao operador de máximo, não sendo possível obter uma solução explícita para as utilidades. Apresentam-se em seguida dois algoritmos iterativos para obter uma estimativa da utilidade $U(x)$ e a regra de decisão óptima $d(x)$: são os algoritmos de *iteração de valor* e de *iteração de política*.

2.5.1 Iteração de valor

O método de iteração de valor consiste em iterar sucessivamente a equação (2.19) calculando a expressão do lado direito modo a obter uma sucessão de estimativas das utilidades $U(x)$ que se aproximam assimptoticamente das verdadeiras utilidades dos estados. Após a convergência, as utilidades estimadas são usadas para determinar a regra de decisão $d(x)$. O algoritmo 1 ilustra em pseudocódigo este procedimento.

Pode provar-se que este algoritmo converge para as utilidades verdadeiras usando o teorema do ponto fixo (ver demonstração no apêndice A). Resumidamente, a demonstração consiste em observar que as utilidades $U(x)$ na equação de Bellman são um ponto fixo. Se a expressão do lado direito da equação de Bellman definir uma contracção, então, pelo teorema do ponto fixo, a iteração dessa expressão converge para um único ponto fixo, que é a solução procurada.

Algoritmo 1 Iteração de valor

1: $U(x) \leftarrow 0$ ▷ Inicialização
2: **repeat** ▷ Estimação das utilidades
3: $\quad \Delta \leftarrow 0$
4: $\quad$ **for all** $x \in \mathcal{X}$ **do**
5: $\quad\quad U'(x) \leftarrow \max_a \left[r(x) + \gamma \sum_{x' \in \mathcal{X}} p(x' \mid x, a')\, U(x') \right]$
6: $\quad\quad \Delta \leftarrow \max\left(\Delta, |U(x) - U'(x)|\right)$
7: $\quad$ **end for**
8: $\quad U(x) \leftarrow U'(x)$
9: **until** $\Delta < \epsilon(1-\gamma)/\gamma$
10: **for all** $x \in \mathcal{X}$ **do** ▷ Determinação da política óptima
11: $\quad d(x) \leftarrow \arg\max_a \left[r(x) + \gamma \sum_{x' \in \mathcal{X}} p(x' \mid x, a')\, U(x') \right]$
12: **end for**

As últimas três linhas do algoritmo, linhas 10–12, são na realidade desnecessárias pois as acções foram implicitamente calculadas na maximização da linha 5. Seria suficiente guardar nessa altura as acções correspondentes ao máximo.

2.5.2 Iteração de política

O algoritmo de iteração de política é um método alternativo, e computacionalmente mais eficiente, ao algoritmo de iteração de valor para obter uma política óptima. A ideia consiste em dividir as iterações em duas partes: numa primeira parte é calculada uma estimativa das utilidades com base numa política provisória π (não óptima); em seguida é recalculada a política provisória assumindo as utilidades calculadas no passo anterior. Estes dois passos são iterados até se obter uma política estável.

Um resumo do método de iteração de política é o seguinte:

1. Inicialização com utilidades $U(x)$ e uma política π arbitrárias;
2. Estimar utilidades com base na política actual π;
3. Recalcular a política π usando as utilidades estimadas no passo 2;
4. Repetir desde 2 até se obter uma política π estável.

A vantagem deste algoritmo face à iteração de valor tem a ver com o facto de não existir um operador de máximo no cálculo da utilidade, o que torna a equação de Bellman numa sistema de equações lineares. Outro ponto importante diz respeito ao cálculo da política: o algoritmo pára quando se obtém uma política estável (significa que a política não varia de iteração para iteração), não sendo necessário conhecer exactamente as utilidades dos estados.

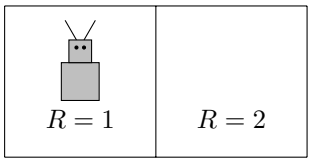
Algoritmo 2 Iteração de política

$U(x) \leftarrow 0$ ▷ Inicialização
$d(x) \leftarrow$ política arbitrária π
repeat
  repeat ▷ Avaliação da política
    $\Delta \leftarrow 0$
    for all $x \in \mathcal{X}$ **do**
      $U'(x) \leftarrow r(x) + \gamma \sum_{x' \in \mathcal{X}} p\big(x' \mid x, d(x)\big)\, U(x')$
      $\Delta \leftarrow \max\big(\Delta, |U(x) - U'(x)|\big)$
    end for
    $U(x) \leftarrow U'(x)$
  until $\Delta < \epsilon$
  $policyStable \leftarrow True$
  for all $x \in \mathcal{X}$ **do** ▷ Melhoramento da política
    $d'(x) \leftarrow \arg\max_a \left[r(x) + \gamma \sum_{x' \in \mathcal{X}} p(x' \mid x, a)\, U(x') \right]$
    if $d(x) \neq d'(x)$ **then**
      $policyStable \leftarrow False$
    end if
    $d(x) \leftarrow d'(x)$
  end for
until $policyStable$

2.6 Exercícios

1. Qual o impacto do factor de desconto no problema? *i.e.*, o que acontece quando γ é maior ou menor?

2. Um robot move-se num espaço dividido em duas regiões. Em cada instante podemos dar dois comandos ao robot: *parado* ou *andar*, que o robot executa correctamente com probabilidade 0.9. Em cada região existe uma recompensa que é recebida pelo robot de cada vez que visita essa região (ver figura abaixo). Determine a política óptima π considerando um horizonte de tempo infinito. Considere um factor de desconto $\gamma = 0.5$.



3. Repita o exercício anterior, mas considerando um custo energético de 1.5 quando é dado o comando de *andar*.

Capítulo 3

Teoria de Jogos (GT)

A teoria de jogos diz respeito a situações de tomada de decisão por mais de um agente decisor, onde se assume que os agentes são racionais. Existem diversas classes de problemas diferentes, com ou sem cooperação. Iremos estudar aqui algumas situações simples de jogos não cooperativos.

3.1 Motivação

Como motivação iremos considerar dois jogos estáticos: o *dilema do prisioneiro* e o *jogo das moedas*. Enquanto no primeiro jogo a melhor estratégia é determinística, no segundo a melhor estratégia é estocástica. Estes jogos serão referidos mais tarde no texto.

3.1.1 Dilema do prisioneiro

Dois indivíduos, suspeitos do mesmo crime, foram presos para interrogatório. Os prisioneiros são interrogados separadamente de modo que nenhum deles tem possibilidade de saber o que o outro responde. É dada a cada um a possibilidade de confessar (C) ou denunciar (D) o outro. O castigo a que cada um está sujeito, em anos de prisão, depende da combinação das respostas de ambos e é dado na tabela seguinte:

	D_2	C_2
D_1	$10, 10$	$0, 15$
C_1	$15, 0$	$2, 2$

Se ambos se denunciarem um ao outro, estratégia (D_1, D_2), obtém-se $(10, 10)$, ou seja 10 anos de prisão para cada um. Por outro lado se um denuncia e o outro confessa, estratégia (D_1, C_2), então o resultado $(0, 15)$ indica que o primeiro sai em liberdade enquanto o segundo é preso 15 anos.

O dilema que se coloca a cada prisioneiro é se deve ou não denunciar o outro. Numa primeira observação da tabela parece que ambos os prisioneiros sairiam melhor com a solução (C_1, C_2), levando apenas 2 anos de cadeia cada um. No entanto, se um prisioneiro suspeita que o outro confessa, o primeiro terá a tendência para denunciar o segundo de modo a sair em liberdade. Ou seja, se cada prisioneiro tentar "safar-se", minimizando os seus anos de cadeia, observamos que será sempre melhor denunciar independentemente da acção do outro. Como resultado obtêm-se

as estratégias (D_1, D_2) e em 10 anos de cadeia para cada um, uma situação claramente pior do que a (C_1, C_2).

3.1.2 Jogo das moedas

Dois jogadores escolhem em segredo uma face (H, T) de uma moeda. Se ambos escolherem a mesma face o jogador 1 ganha a moeda do adversário ficando com as duas moedas. Se escolherem faces diferentes o jogador 2 ganha a moeda do jogador 1.

	H_2	T_2
H_1	$+1, -1$	$-1, +1$
T_1	$-1, +1$	$+1, -1$

Em cada jogo, um jogador ganha a moeda (+1) que o outro jogador perde (-1). A este tipo de jogos dá-se o nome de *jogos de soma zero* (*zero-sum games*), ou seja, são jogos em que o proveito de um é o prejuízo do outro. Neste jogo em particular, as melhores estratégias são cada jogador jogar aleatoriamente com probabilidades $(1/2, 1/2)$. Se ambos jogarem com esta estratégia, a recompensa esperada é nula. Caso um jogador não siga esta estratégia, o seu adversário pode tirar proveito disso e obter uma recompensa esperada positiva.

3.2 Jogos estáticos

Um jogo estático é um jogo em que existe apenas um instante de decisão. Nesse instante, cada agente decide uma estratégia de jogo (*i.e.*, a acção que vai tomar) independentemente dos restantes agentes.

Em jogos estáticos não há uma distinção clara entre estratégias e acções. Havendo apenas um instante de decisão, uma estratégia apenas especifica uma acção a tomar e o conjunto das estratégias disponíveis para um agente é simplesmente o conjunto das acções disponíveis para esse agente.

De um modo geral, na caracterização de um jogo estático é necessário identificar os seguintes elementos:

- Agentes de decisão (os jogadores), enumerados por $\{1, 2, \ldots, n\}$;
- O conjunto $\mathcal{A}_i$ das acções disponíveis para cada agente i (as acções podem ter natureza discreta ou contínua);
- As recompensas $r_i : \mathcal{A}_1 \times \cdots \times \mathcal{A}_n \to \mathbb{R}$ de cada agente i, como função das acções tomadas por todos os agentes.

As recompensas são especificadas por n funções

$$r_i(a_1, \ldots, a_n), \quad i = 1, \ldots, n, \tag{3.1}$$

onde os argumentos são as acções tomadas pelos agentes $1, \ldots, n$. Uma forma de descrever estas funções é numa tabela (possivelmente multidimensional) como a do dilema do prisioneiro. Um jogo descrito numa tabela diz-se que está na *forma normal*.

Num jogo estático, uma estratégia pura consiste simplesmente em especificar a acção a tomar. Alternativamente, podemos definir uma estratégia mista para um agente i por intermédio de uma distribuição de probabilidade

$$p(a_i \,|\, \sigma_i), \tag{3.2}$$

onde $a_i \in \mathcal{A}_i$ são as acções disponíveis e σ_i especifica a distribuição de probabilidade usada pelo agente i. A execução desta estratégia consiste em tomar uma acção aleatória com a distribuição de probabilidade dada. O conjunto de todas as estratégias mistas para o agente i é representado por Σ_i.

Se os agentes $1, \ldots, n$ usam estratégias mistas $\sigma_1, \ldots, \sigma_n$ então as recompensas obtidas são variáveis aleatórias. A recompensa esperada pelo i-ésimo agente é

$$\begin{aligned} r_i(\sigma_1, \ldots, \sigma_n) &= \sum_{a_1,\ldots,a_n} p(a_1, \ldots, a_n|\sigma_1, \ldots, \sigma_n) r_i(a_1, \ldots, a_n) \\ &= \sum_{a_1,\ldots,a_n} p(a_1|\sigma_1) \cdots p(a_n|\sigma_n) r_i(a_1, \ldots, a_n). \end{aligned} \tag{3.3}$$

3.2.1 Estratégias dominadas

Comparando várias estratégias, podemos "resolver" um jogo mantendo as melhores estratégias e eliminando as piores.

Definição 1 (Estratégia estritamente dominada)**.** *Para um agente i, uma estratégia σ_i é estritamente dominada por outra σ_i' se*

$$r_i(\ldots, \sigma_i, \ldots) < r_i(\ldots, \sigma_i', \ldots) \tag{3.4}$$

para todas as outras estratégias de todos os outros agentes.

Isto é, para o agente i é sempre pior usar a estratégia σ_i do que a σ_i'. Concluímos que a estratégia estritamente dominada σ_i não deve ser usada e portanto pode ser eliminada.

No dilema do prisioneiro, as estratégias C são estritamente dominadas pelas D para ambos os prisioneiros, pelo que as estratégias C podem ser eliminadas. A solução é resultante é (D, D). Admitindo estratégias mistas, a estratégia com probabilidades $\sigma_1 = (0.5, 0.5)$ é estritamente dominada pela estratégia $\sigma_2 = (0.6, 0.4)$ (verifique!). Podemos até verificar que, neste jogo, todas as estratégias mistas são dominadas pela estratégia pura D, pelo que se as eliminarmos ficamos novamente com a solução (D, D).

A resolução de um jogo por esta via chama-se *eliminação iterada de estratégias estritamente dominadas*. O resultado obtido é independente da ordem pela qual as estratégias são eliminadas.

Nem todos os jogos podem ser resolvidos por eliminação de estratégias estritamente dominadas, simplesmente porque estas podem não existir.

Definição 2 (Estratégia fracamente dominada)**.** *Para um agente i, uma estratégia σ_i é fracamente dominada por outra σ_i' se*

$$r_i(\ldots, \sigma_i, \ldots) \leq r_i(\ldots, \sigma_i', \ldots) \tag{3.5}$$

para todas as estratégias de todos os outros agentes e existe pelos menos uma estratégia adversária σ_j' onde σ_i é estritamente pior

$$r_i(\ldots, \sigma_i, \ldots, \sigma_j', \ldots) < r_i(\ldots, \sigma_i', \ldots, \sigma_j', \ldots). \tag{3.6}$$

A condição adicional (3.6) impede que duas estratégias com recompensas exactamente iguais sejam consideradas como fracamente dominadas uma da outra.

Exemplo 7. *O jogo*

	C	*D*
A	*3,3*	*2,2*
B	*2,1*	*2,1*

não tem estratégias estritamente dominadas mas tem estratégias fracamente dominadas: a estratégia D é fracamente dominada pela C e a estratégia B é fracamente dominada pela A. A eliminação de estratégias fracamente dominadas leva à solução (A, C).

Exemplo 8. *Num jogo com n jogadores, cada um escolhe em segredo um número inteiro de 0 a 100. No final, calcula-se a média μ de todas as respostas e ganha o/os jogadores mais perto de $\mu/2$. Que número deveremos escolher?*

Este problema pode ser resolvido por eliminação iterada de estratégias fracamente dominadas. Para o jogador i, não faz sentido apostar num número maior que 50, pois $\mu/2$ nunca será superior a 50 (apostar 50 estará sempre mais perto de $\mu/2$ do que 51). Assim, o jogador i pode eliminar todas as estratégias > 50. Mas o mesmo irão fazer todos os outros jogadores: ninguém irá apostar em números > 50 (pensa o jogador i), o que implica que $\mu/2 \leq 25$. O jogador i conclui então que as estratégias > 25 podem ser eliminadas. Mas o mesmo irão fazer todos os outros jogadores...

Continuando este reciocínio ad infinitum, *o jogador i chega à conclusão que deverá apostar 0 (zero). E à mesma conclusão chegam todos os outros jogadores!*

Neste exemplo assumiu-se que todos os jogadores são racionais e que, durante o seu raciocínio, se colocam no papel dos adversários assumindo que estes são igualmente racionais (e que eles por sua vez se vão colocar no papel dos outros, e por aí fora). Isto é, colocam-se todos no papel uns dos outros infinitas vezes.

(Nota: Experimentando com seres humanos observa-se que não somos totalmente racionais e que não respondemos todos "zero". No entanto, como a nossa análise não se resume aos seres humanos, vamos assumir sempre racionalidade. Se o jogo fosse jogado por n robots perfeitamente racionais, todos iriam apostar "zero".)

Ao contrário do que acontece com a eliminação de estratégias estritamente dominadas, o resultado obtido pela eliminação de estratégias fracamente dominadas pode depender da ordem pela qual a eliminação é efectuada. Por exemplo, no jogo

	C	D	E
A	10,0	5,1	4,0
B	10,1	5,0	1,-1

qualquer um dos dois jogadores tem estratégias fracamente dominadas, nomeadamente B e E, pelo que podemos começar por eliminar por qualquer uma delas:

- Eliminando pela ordem $B \to E \to C$ obtém-se a solução (A, D);
- Eliminando primeiro E, não fica mais nenhuma estratégia dominada, nem estritamente nem fracamente, e obtêm-se quatro soluções: (A, C), (A, D), (B, C), (B, D).

3.2.2 Equilíbrio de Nash

Muitos jogos não têm estratégias dominadas pelo que, não havendo eliminação de estratégias, ficamos com uma solução trivial em que todas as estratégias são solução. Por exemplo, o jogo

	C	D	E
A	0,3	2,2	3,-1
B	2,3	1,1	$\boxed{4,4}$

não tem estratégias dominadas, mas pode observar-se que a estratégia (B, E) é claramente melhor que as outras. Se aplicarmos o princípio de "colocarmo-nos no papel dos outros", repetidamente, chegamos sempre à conclusão que (B, E) é a melhor estratégia:

- Colocando-nos no papel do agente 1, vamos supor que usamos a estratégia A; então o agente 2 irá usar C; mas nesse caso deveremos antes usar a B; mas então o agente 2 irá usar a E e nós iremos manter B. Chegámos à solução (B, E). Se tivéssemos antes começado pela estratégia B o raciocínio teria sido encurtado mas a conclusão seria a mesma.
- Colocando-nos no papel do agente 2, a sequência seria $C \to B \to E$, ou $D \to A \to C \to B \to E$, ou $E \to B \to E$. Em qualquer dos casos chegamos sempre a (B, E), e ficamos lá!

O raciocínio que se aplicou no exemplo anterior consiste em determinar, em cada passo, a *melhor resposta* às estratégias do adversário.

Definição 3 (Melhor resposta)**.** *Para o agente i, e dada uma certa combinação de estratégias dos adversários*

$$\sigma_{-i} \triangleq (\sigma_1, \ldots, \sigma_{i-1}, \sigma_{i+1}, \ldots, \sigma_n), \tag{3.7}$$

uma estratégia $\hat{\sigma}_i$ é uma melhor resposta se

$$\hat{\sigma}_i \in \arg\max_{\sigma_i \in \Sigma_i} r_i(\sigma_1, \ldots, \sigma_i, \ldots, \sigma_n). \tag{3.8}$$

Note que pode existir um conjunto de melhores respostas para mesma combinação (3.7) de estratégias dos adversários, donde resulta o sinal de pertence $\in$ em vez de uma igualdade em (3.8).

Se assumirmos que todos os agentes usam as suas melhores respostas e colocam-se no papel uns dos outros, repetidamente, chegamos eventualmente a uma solução de equilíbrio onde todos usam a sua melhor resposta.

Definição 4 (Equilíbrio de Nash)**.** *Diz-se que um conjunto de estratégias $(\sigma_1^*, \ldots, \sigma_n^*)$ é um equilíbrio de Nash se*

$$\forall i \in \{1, \ldots, n\}: \quad \sigma_i^* \in \arg\max_{\sigma_i \in \Sigma_i} r_i(\sigma_1^*, \ldots, \sigma_{i-1}^*, \sigma_i, \sigma_{i+1}^*, \ldots, \sigma_n^*). \tag{3.9}$$

A condição de equilíbrio de Nash diz simplesmente que cada estratégia em $(\sigma_1^*, \ldots, \sigma_n^*)$ é uma melhor resposta às restantes.

Exemplo 9. *No dilema do prisioneiro as estratégias (D_1, D_2) são as melhores respostas uma à outra e portanto são um equilíbrio de Nash. As estratégias (C_1, C_2) não são um equilíbrio de Nash, pois a melhor resposta a C_2 é D_1 e a melhor resposta a C_1 é D_2.*

Exemplo 10. *No jogo das moedas não há nenhum par de estratégias puras que seja um equilíbrio de Nash, mas há um equilíbrio de Nash com estratégias mistas. Vamos supor que o agente 1 decide por H ou T com probabilidades $\sigma_1 = (p, 1-p)$, o agente 2 fazendo o mesmo com probabilidades $\sigma_2 = (q, 1-q)$. As recompensas esperadas são*

$$\begin{aligned} r_1(\sigma_1, \sigma_2) &= pq - p(1-q) - (1-p)q + (1-p)(1-q) \\ &= (4q-2)p + 1 - 2q, \end{aligned} \tag{3.10}$$

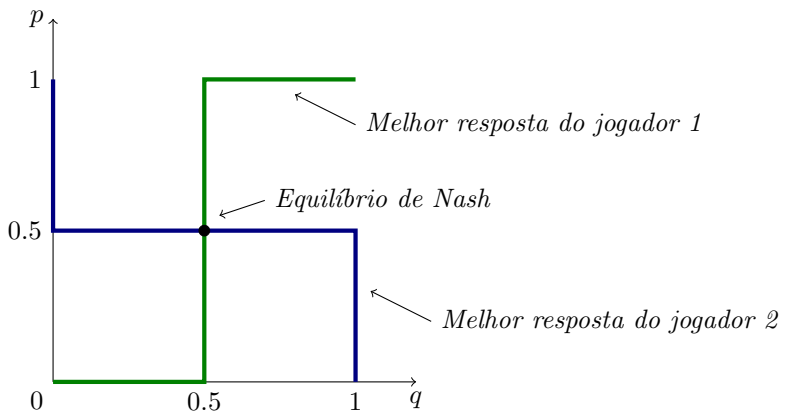
$$\begin{aligned} r_2(\sigma_1, \sigma_2) &= -pq + p(1-q) + (1-p)q - (1-p)(1-q) \\ &= (-4p+2)q + 2p - 1. \end{aligned} \tag{3.11}$$

Para o jogador 1, a melhor resposta depende do sinal de $(4q-2)$*:*

$$\hat{\sigma}_1 = \begin{cases} (0,1) & \text{se } q < 1/2 \\ \text{qualquer} & \text{se } q = 1/2 \\ (1,0) & \text{se } q > 1/2 \end{cases} \quad \Rightarrow \quad r_1(\hat{\sigma}_1, \sigma) = \begin{cases} 1-2q & \text{se } q < 1/2 \\ 0 & \text{se } q = 1/2 \\ 2q-1 & \text{se } q > 1/2 \end{cases}$$

Para o jogador 2 constroem-se melhores respostas de forma semelhante.

Podemos representar graficamente as melhores respostas de cada jogador em função da estratégia do outro:



O equilíbrio de Nash acontece na intersecção das curvas $(0.5, 0.5)$*, que é o único ponto em que ambos os jogadores usam as suas melhores respostas.*

Exercício 4 (Slotted ALOHA simplificado)**.** *No sistema de comunicação Slotted ALOHA, vários dispositivos de rede podem transmitir em slots de tempo de tamanho fixo bem conhecido. Os dispositivos não verificam se estão comunicações em curso, pelo que podem existir colisões. Em caso de colisão retransmitem no slot seguinte com probabilidade* p *ou esperam por mais tarde com probabilidade* $1-p$*.*

Neste exercício vamos considerar uma versão simplificada (e distorcida) com apenas dois dispositivos de rede e apenas dois slots *de tempo utilizáveis após a colisão. A tabela de recompensas é a seguinte:*

	Enviar	*Esperar*
Enviar	*0, 0*	*1, 0.5*
Esperar	*0.5, 1*	*0.5, 0.5*

Calcule os equilíbrios de Nash e as respectivas recompensas esperadas.

Na definição 4, o equilíbrio de Nash foi definido em termos das melhores respostas dos agentes. No entanto, o equilíbrio de Nash pode equivalentemente ser escrito em termos das recompensas.

Definição 5 (Equilíbrio de Nash)**.** *Um equilíbrio de Nash é uma tupla de estratégias* $(\sigma_1^*, \ldots, \sigma_n^*)$ *tal que, para toda os agentes* $i \in \{1, \ldots, n\}$*,*

$$r_i(\sigma_1^*, \ldots, \sigma_n^*) \geq r_i(\sigma_1^*, \ldots, \sigma_{i-1}^*, \sigma_i, \sigma_{i+1}^*, \ldots, \sigma_n^*). \tag{3.12}$$

Isto é, para cada agente i*, não há nenhuma estratégia que leve a uma recompensa (esperada) estritamente maior do que a obtida com* σ_i^**, mantendo fixas as estratégias adversárias.*

Em todos os problemas considerados até ao momento encontramos sempre equilíbrios de Nash (nem que fosse usando estratégias mistas). Será que existem sempre equilíbrios de Nash?

Teorema 1 (Nash)**.** *Todos os jogos com um número finito de agentes e estratégias tem pelo menos um equilíbrio de Nash.*

Alguns jogos têm mais do que um equilíbrio de Nash. Certos jogos têm apenas um número ímpar de equilíbrios: os jogos em que uma pequena perturbação nas recompensas não leva à extinção ou criação de equilíbrios de Nash têm esta propriedade (essencialmente significa que não há igualdades na equação (3.12)).

3.3 Exercícios

1. O jogo "Pedra-Papel-Tesoura" é um jogo muito simples em que duas pessoas fazem em simultâneo um gesto com uma mão, representando uma pedra (mão fechada), papel (mão aberta) ou tesoura (dedos indicador e médio esticados). Ganha o jogador cujo objecto for mais forte: pedra ganha à tesoura, tesoura ganha ao papel, e papel ganha à pedra.

   (a) Represente este jogo na forma normal.

   (b) O jogo tem estratégias estritamente ou fracamente dominadas?

   (c) Determine o equilíbrio de Nash do jogo.

Apêndice A

Convergência da iteração de valor

A demonstração consiste essencialmente em provar que uma iteração de valor é uma contracção num espaço métrico apropriado, garantindo-se então a existência e unicidade de um ponto fixo pelo teorema do ponto fixo de Banach.

Iniciaremos por introduzir o teorema do ponto fixo e depois aplicá-lo neste problema.

Definição 6 (Ponto fixo)**.** *Diz-se que x é um ponto fixo de uma função f se*

$$x = f(x). \tag{A.1}$$

Uma função f pode ter zero, um, ou vários pontos fixos.

Exemplo 11. *A função $f(x) \triangleq x^2$ tem como pontos fixos as soluções de*

$$x = x^2. \tag{A.2}$$

Esta equação tem como soluções $x = 0$ e $x = 1$, portanto os pontos fixos são $\{0, 1\}$.

Definição 7 (Contracção)**.** *Uma função f definida num espaço métrico diz-se que é uma contracção se existe uma constante $k < 1$ tal que*

$$d(f(x), f(y)) \leq kd(x, y). \tag{A.3}$$

O conceito de contracção pode ser interpretado geometricamente como ilustrado na figura A.1. Se uma função f satisfaz a propriedade (A.3) em todo o espaço, então pode ser aplicada aos pontos $f(x)$ e $f(y)$ obtendo-se

$$d\big(f(f(x)), f(f(y))\big) \leq kd(f(x), f(y)), \tag{A.4}$$

e assim sucessivamente. Observamos que, à medida que a função f é iterada, a distância vai reduzindo-se de um factor de pelo menos k, eventualmente convergindo para um ponto fixo. É este o resultado do teorema do ponto fixo que se enuncia em seguida sem demonstração.

Teorema 2 (Teorema do ponto fixo de Banach)**.** *Seja (X, d) um espaço métrico completo*[1] *e f uma contracção. Então f tem um único ponto fixo x^*, isto é, $x^* = f(x^*)$.*

O ponto fixo x^* pode ser encontrado começando com um ponto arbitrário x_0 e definindo uma sucessão iterativamente por

$$x_n = f(x_{n-1}), \qquad n = 1, 2, 3, \ldots \tag{A.5}$$

Esta sucessão é convergente e o seu limite é o ponto fixo x^*.

[1] Um espaço métrico completo é um espaço com uma métrica $d(\cdot, \cdot)$ e que inclui os limites de todas as sucessões (de Cauchy) que se podem formar nesse espaço.

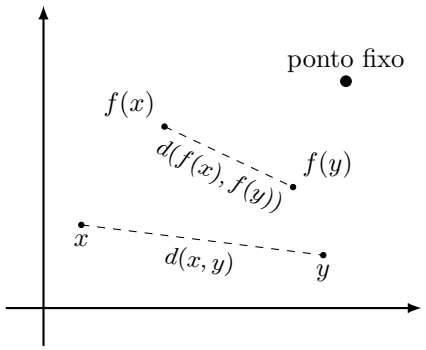


Figura A.1: Exemplo de uma contracção. A distância entre $f(x)$ e $f(y)$ é menor que a distância entre x e y.

Exemplo 12. *Com a métrica* $d(x, y) \triangleq |x - y|$*, a função* $f(x) \triangleq x/2$ *é uma contracção pois*

$$d\big(f(x), f(y)\big) = d\left(\frac{x}{2}, \frac{y}{2}\right) = \left|\frac{x}{2} - \frac{y}{2}\right| \leq k|x - y| = kd(x, y), \tag{A.6}$$

para $k \leq 0.5$*. O único ponto fixo é obtido tomando o limite* $n \to \infty$ *da sucessão* x_n *definida em* (A.5)*. Obtemos neste caso que*

$$x^* = \lim_{n\to\infty} x_n = \lim_{n\to\infty} \frac{1}{2}x_{n-1} = \lim_{n\to\infty} \frac{1}{2^2}x_{n-2} = \cdots = \lim_{n\to\infty} \frac{1}{2^n}x_0 = 0. \tag{A.7}$$

O ponto fixo é $x^* = 0$*.*

Regressamos agora ao problema de provar a convergência do método de iteração de valor. A ideia é provar que uma iteração do método é uma contracção relativamente à métrica

$$d(x, y) = \max_i |x_i - y_i|. \tag{A.8}$$

A distância entre dois pontos x e y é a maior diferença entre as coordenadas dos pontos. Esta não é a distância euclideana usual, mas satisfaz os axiomas de uma distância:

$$d(x, y) \geq 0, \text{ com igualdade sse } x = y \qquad \text{(não negativa)} \tag{A.9}$$
$$d(x, y) = d(y, x) \qquad \text{(simétrica)} \tag{A.10}$$
$$d(x, y) \leq d(x, z) + d(z, y) \qquad \text{(desigualdade triangular)} \tag{A.11}$$

Considere-se o vector das utilidades U, em que cada componente é a utilidade $U(s)$ do s-ésimo estado, e define-se a função f como sendo uma iteração do método de iteração de valor.

O princípio de optimalidade de Bellman consiste em encontrar o U^* tal que $U^* = f(U^*)$, ou seja encontrar o ponto fixo de f. Se f é uma contracção, então, pelo teorema do ponto fixo de Banach, iterando $U_n = f(U_{n-1})$ obtém-se uma sucessão de vectores de utilidade U_n que converge para o ponto fixo U^*.

Para provar que f é uma contracção consideram-se dois vectores de utilidade arbitrários U e

V. Verifica-se que

$$\begin{aligned}
d\big(f(U), f(V)\big) &= \max_s \big|f(U) - f(V)\big| \\
&= \max_s \left| \max_a \gamma \sum_{s' \in \mathcal{S}} P(s,a,s')U(s') - \max_a \gamma \sum_{s' \in \mathcal{S}} P(s,a,s')V(s') \right| \\
&= \gamma \max_s \left| \max_a \left(\sum_{s' \in \mathcal{S}} P(s,a,s')U(s') - \sum_{s' \in \mathcal{S}} P(s,a,s')V(s') \right) \right| \\
&= \gamma \max_s \left| \max_a \sum_{s' \in \mathcal{S}} P(s,a,s')\big(U(s') - V(s')\big) \right| \\
&\leq \gamma \max_{s'} \big|U(s') - V(s')\big| \\
&= \gamma\, d(U,V),
\end{aligned}$$

pelo que uma iteração f é uma contracção.