Éléments de processus stochastiques

Méthodes de Monte Carlo par chaines de Markov - application à la cryptanalyse

Profs. Louis Wehenkel et Pierre Geurts
Assistante : Laurine Duchesne

Année académique 2017-2018

Le travail sera réalisé par groupe de trois étudiants. Les consignes générales pour réaliser le travail, et celles relatives à la manière de rédiger les rapports et préparer la présentation orale, sont les mêmes pour tous les projets et sont communiquées par le Professeur V. Denoël, coordinateur du cours.

Des éléments complémentaires de théorie nécessaires pour réaliser ce projet seront expliqués lors des séances d'encadrement du travail. Nous encourageons vivement les étudiants à se documenter eux-mêmes sur les sujets abordés dans ce projet. Des pointeurs vers des sources de théorie/pratique seront fournis au fur et à mesure de la réalisation du travail.

Le travail sera encadré au moyen de séances de questions/réponses qui seront programmées au fur et à mesure de l'avancement du projet. Ces séances auront lieu les mardis de 10h45 à 12h45 au local 01 (B37). Seules les questions posées lors de ces séances donneront lieu à des réponses de la part des encadrants.

Contexte général et objectifs

L'objectif général du projet est de développer un algorithme permettant de déchiffrer un message crypté par substitution monoalphabétique. Pour construire ce système, on se basera sur la méthode de Monte Carlo par chaines de Markov (MCMC pour *Markov chain Monte Carlo* en anglais).

Le projet est divisé en deux parties. La première partie du projet, relativement théorique, a pour but de vous familiariser avec la modélisation du langage via des chaines de Markov ainsi qu'avec la méthode MCMC. La deuxième partie, qui constitue le cœur du projet, vise à mettre en œuvre concrètement la méthode MCMC pour implémenter le système de décryptage de textes chiffrés.

1 Première partie : chaines de Markov pour la modélisation du langage et MCMC

Dans cette première partie du projet, on se propose d'aborder la modélisation du langage par une chaine de Markov ainsi que l'étude de l'algorithme MCMC.

Définitions et notations

Dans cette section, nous fournissons les définitions et notations minimales nécessaires pour la première partie du projet. Nous vous encourageons néanmoins à consulter d'autres références pour obtenir plus de détails à propos de la méthode MCMC.

Chaines de Markov. Soit une suite de variables aléatoires $\{X_1, X_2, \ldots, X_t, \ldots\}$. Cette suite définit un modèle (ou une chaine) de Markov d'ordre 1 ssi, pour tout $t \geq 1$, la distribution conjointe de t premières variables peut se factoriser comme suit :

$$P(X_1, X_2, \ldots, X_t) = P(X_1) \prod_{l=2}^{t} P(X_l | X_{l-1}).$$

Dans cette partie, on supposera que le modèle de Markov est discret et on notera sans restriction $\{1, \ldots, N\}$ l'ensemble des valeurs possibles des variables X_i, appelées les états.

Méthode de Monte Carlo. La méthode de Monte Carlo de base permet d'estimer la valeur de l'espérance $E\{h(X)\}$ d'une fonction h d'une variable aléatoire X en utilisant la moyenne empirique

$$\hat{I} = \frac{1}{n} \sum_{i=1}^{n} h\left(x^{(i)}\right),$$

où les $x^{(i)}$ ont été générés aléatoirement *i.i.d.* selon la densité p_X. Lorsque que la variance de $h(X)$ est finie, $\hat{I}$ converge presque sûrement vers $E\{h(X)\}$, avec une erreur standard qui décroit en $1/\sqrt{n}$ avec la taille de l'échantillon. Des versions plus générales de la loi des grands nombres permettent de garantir la convergence de cette méthode dans certains cas où les $x^{(i)}$ ne sont pas indépendantes ou lorsque la variance de $h(X)$ n'est pas finie.

Méthodes MCMC. Les méthodes MCMC consistent à simuler une chaine de Markov ergodique dont la distribution stationnaire est égale à p_X. Dans ce projet, nous utiliserons l'algorithme de Metropolis-Hastings. Cet algorithme utilise une fonction q appelée *densité de proposition* qui, en pratique, est telle qu'il est possible de générer des échantillons selon cette densité.

Algorithm 1 Algorithme de Metropolis-Hastings.

Set starting state $x^{(0)}$,$t = 1$
while convergence not reached **do**
  Generate $y^{(t)} \sim q(y|x^{(t-1)})$
  $\alpha \leftarrow \min\left\{1, \frac{p_X(y^{(t)})}{p_X(x^{(t-1)})} \frac{q(x^{(t-1)}|y^{(t)})}{q(y^{(t)}|x^{(t-1)})}\right\}$
  Generate u uniformly in $[0, 1[$
  if $u < \alpha$ **then**
    $x^{(t)} \leftarrow y^{(t)}$
  else
    $x^{(t)} \leftarrow x^{(t-1)}$
  end if
  $t \leftarrow t + 1$
end while

1.1 Chaine de Markov pour la modélisation du langage

Claude Shannon fut le premier à proposer de modéliser le langage via des chaines de Markov. Dans ce travail, nous allons modéliser la séquence de lettres, symboles de ponctuation et espace de la langue anglaise par une chaine de Markov d'ordre 1. Pour se familiariser avec ce modèle, commençons par un alphabet composé uniquement de 4 lettres a, b, c et d.

Questions :

1. Soit la séquence `seq1` composée uniquement des quatre lettres a, b, c et d et fournie dans le fichier Matlab `seq1.m`. Utilisez la méthode du maximum de vraisemblance pour construire la matrice de transition Q (avec $[Q]_{i,j} = \mathbb{P}(X_{t+1} = j|X_t = i)$) ainsi que la distribution de probabilité initiale π_0 d'une chaine de Markov modélisant cette séquence. Représentez le diagramme d'états de la chaine de Markov correspondante.
2. Sur base de la matrice de transition Q estimée au point précédent, calculez les quantités suivantes pour des valeurs de t croissantes :
   — $\mathbb{P}(X_t = x)$, où $x = a, b, c, d$, en supposant que la première lettre est choisie au hasard, i.e. la lettre initiale est tirée dans une loi uniforme discrète.
   — $\mathbb{P}(X_t = x)$, où $x = a, b, c, d$, en supposant que la première lettre est toujours c,
   — Q^t, c'est-à-dire la t-ième puissance de la matrice de transition.
   Représentez l'évolution des deux premières grandeurs sur un graphe. Discutez et expliquez les résultats obtenus sur base de la théorie.
3. En déduire la distribution stationnaire π_∞ de la chaine de Markov définie par $[\pi_\infty]_j = \lim_{t\to\infty} \mathbb{P}(X_t = j)$.
4. Générez une réalisation aléatoire de longueur T de la chaine de Markov en démarrant d'une lettre choisie aléatoirement selon la distribution stationnaire. Calculez pour chaque lettre le nombre de fois qu'elle apparait dans la réalisation rapporté à la longueur de la réalisation. Observez l'évolution de ces valeurs pour chaque lettre lorsque T croît.
5. Que concluez-vous de cette expérience ? Reliez ce résultat à la théorie.

1.2 Algorithme MCMC

Les méthodes MCMC sont souvent utilisées lorsqu'il est difficile d'échantillonner directement selon une distribution p_X. Par exemple en inférence bayésienne, on souhaite typiquement générer des échantillons d'un paramètre θ selon sa densité a posteriori $p_{\theta|D}$, alors que celle-ci n'est généralement connue qu'à un facteur de normalisation près.

Les méthodes MCMC peuvent être appliquées aussi bien à des variables continues que discrètes. Cependant, dans le cadre de ce travail nous nous focalisons sur l'application de cette méthode dans le cas où la variable cible est discrète, et nous nous limiterons à l'étude de l'algorithme de Metropolis-Hastings.

Questions :

1. Etant donné une matrice de transition Q et une distribution initiale π_0 d'une chaine de Markov invariante dans le temps qui satisfont les équations de balance détaillée (c'est-à-dire $\forall i, j \in \{1, \ldots, N\}$, $\pi_0(i)[Q]_{i,j} = \pi_0(j)[Q]_{j,i}$), montrez que π_0 est une distribution stationnaire de la chaine de Markov. Dans quel(s) cas celle-ci est-elle unique ?
2. Dans le cas où X est discrète, démontrer que l'application de l'algorithme de Metropolis-Hastings en remplaçant p_X par une fonction f telle que $\forall x : f(x) = cp_X(x)$, où c est une constante, génère une chaine de Markov qui satisfait les équations de balance détaillée avec $p_X(x)$ comme distribution stationnaire. Quelles autres conditions la chaine doit-elle respecter pour que l'algorithme de Metropolis-Hastings fonctionne ?
   Indice : commencez par écrire les probabilités de transition en prenant en compte les différents cas possibles (rejet ou acceptation) et vérifiez que les équations de balance détaillée sont satisfaites.

2 Deuxième partie : décryptage d'une séquence codée

L'objectif de cette seconde partie est de mettre au point un système de décryptage d'une sequence codée par substitution en se basant sur les techniques étudiées dans la première partie. Le cryptage par substitution monoalphabétique consiste simplement à remplacer chaque lettre par une autre de l'alphabet, en s'assurant, pour permettre le décryptage, qu'il n'y a pas deux lettres distinctes remplacées par la même lettre. Un code de substitution particulier est donc défini par une permutation particulière des lettres de l'alphabet.

Un texte en langue anglaise T peut être vu comme une séquence de caractères générée par une chaine de Markov caractérisée par une distribution initiale π_0 et une matrice de transition Q. Notons par θ_* le code de substitution, T_* le texte de départ, et par $D = \theta_*(T_*)$ la version chiffrée du texte à disposition du cryptanalyste.

Afin de retrouver une approximation $\hat{T}$ du texte T_* à partir de D, une solution serait de considérer chaque permutation possible de l'alphabet, de l'appliquer (en sens inverse) au texte encodé pour obtenir $T' = \theta^{-1}(D)$ et de garder la séquence qui maximise la vraisemblance $\mathcal{L}(T', \pi_0, Q)$ selon le modèle de chaine de Markov utilisé pour modéliser la langue anglaise. Compte tenu du nombre de caractères dans la langue anglaise (40, si on se limite aux minuscules) le nombre $|\Theta|$ de permutations candidates possibles vaut environ 10^{48}, ce qui fait que cette solution n'est pas utilisable en pratique.

Une autre possibilité serait de tirer aléatoirement des permutations dans l'ensemble Θ comprenant toutes les permutations possibles, selon la loi a posteriori résultant du texte

chiffré D et du modèle de Markov de la langue anglaise, et de déterminer le mode θ_{MAP} de cette loi pour estimer θ_* et puis T_*. La probabilité a posteriori d'une permutation $\theta \in \Theta$ est donnée par :

$$p_{\theta|D,\pi_0,Q}(\theta) = \frac{\mathcal{L}(\theta^{-1}(D), \pi_0, Q) p_\theta(\theta)}{\sum_{\theta \in \Theta} \mathcal{L}(\theta^{-1}(D), \pi_0, Q) p_\theta(\theta)},$$

où $p_\theta(\theta)$ représente une distribution de probabilité a priori modélisant le choix du code par celui qui a chiffré le texte.

L'objectif de cette partie du travail sera d'utiliser l'algorithme de Metropolis-Hastings pour échantillonner selon cette distribution $p_{\theta|D,\pi_0,Q}$ afin de décoder une séquence D donnée. L'alphabet de caractères ainsi que la matrice de transition de la chaine de Markov d'ordre 1 modélisant le langage sont fournis sur la page web du cours. Chaque groupe sera en charge de décoder un texte D qui lui sera fourni.

Questions :

1. Montrez comment calculer la cardinalité de l'ensemble Θ de tous les codes possibles.
2. Expliquez comment calculer la vraisemblance selon le modèle π_0, Q d'un texte T' non chiffré et puis celle associée à un texte D résultant du chiffrement par un code θ donné.
3. En supposant que la loi a priori $p_\theta(\theta)$ vaut $1/10$ pour chaque code parmi un petit ensemble de 10 codes candidats et 0 pour tous les autres codes, décrivez un algorithme permettant de faire le tirage directement selon $p_{\theta|D,\pi_0,Q}$ et de déchiffrer le texte.
4. Utilisez MatLab pour implémenter l'algorithme de Metropolis-Hastings pour tirer θ selon $p_{\theta|D,\pi_0,Q}$ pour une distribution a priori $p_\theta(\theta)$ uniforme sur l'ensemble Θ de tous les codes de substitution possibles sur l'alphabet fourni et le modèle de langage fourni, mais avec une distribution de proposition quelconque q. Décrivez clairement votre implémentation.
5. Choisissez une distribution de proposition q et discutez de la convergence de l'algorithme vers la distribution stationnaire voulue en fonction de ce choix. Testez votre implémentation pour déchiffrer des textes en anglais de longueur croissante encodés par un code de substitution que vous fixez une fois pour toutes. Proposez une métrique pour analyser la convergence de l'algorithme et utilisez-la pour étudier l'impact de la longueur du texte à décoder sur la convergence et la qualité du décodage.
6. Etudiez l'impact du choix de la distribution q sur la convergence de la méthode, et choisissez-en une pour vous aider à décrypter le texte qui vous a été fourni. Fournissez la version décryptée du texte en expliquant comment vous l'avez finalement choisie. *Suggestion : il peut être intéressant d'étudier une distribution q indépendante de l'état précédent, c'est-à-dire telle que* $q(y|x) = q(y)$.

3 Bonus (facultatif)

Si vous êtes intéressés, deux questions supplémentaires vous sont proposées en bonus :

— Trouvez une méthode de chiffrement que l'algorithme que vous avez mis au point ne pourra décoder ;

— Adaptez l'algorithme de façon à ce qu'il puisse également déchiffrer un message encodé avec la méthode déterminée au point précédent.

4 Rapport et code

Vous devez nous fournir un rapport contenant vos réponses, concises mais précises, aux questions posées ainsi que les scripts MatLab que vous avez utilisés pour y répondre.

5 Références

Toutes les références, les données, et les codes relatifs au projet seront collectés sur la page Web des projets (`http://www.montefiore.ulg.ac.be/~lduchesne/stocha/`). Une foire aux questions sera également ajoutée sur cette page et complétée au fur et à mesure de l'avancement du projet. Veillez à consulter régulièrement cette page.