Temperature

April 19, 2019

1 Température

Marc Lorenzi
19 avril 2019

```
In [1]: import matplotlib.pyplot as plt
        import random
        import math
```

```
In [2]: plt.rcParams['figure.figsize'] = (8, 8)
```

1.1 1. Un jeu de hasard

1.1.1 1.1 À quoi allons-nous jouer ?

Soit S une liste de taille N. Soit $q \in \mathbb{N}$. On initialise tous les éléments de S à la valeur q, puis on itère l'opération suivante :

On choisit au hasard un entier i entre 0 et $N-1$. Si $S[i] > 0$, - On choisit au hasard j entre 0 et $N-1$. - On augmente $S[j]$ de 1. - On diminue $S[i]$ de 1.

On fait cela un "grand" nombre de fois puis on regarde l'état de la liste S. Combien de cases à 0 ? Combien à 1 ? etc.

L'idée est de modéliser (très naïvement) un système composé de N particules. Au départ, toutes ces particules ont une énergie qE_0 où $E_0 > 0$ est l'énergie d'un "quantum d'énergie". Chaque particule peut interagir avec une autre particule en perdant un quantum d'énergie et en donnant ce quantum à l'autre particule.

1.1.2 1.2 Jouons !

La fonction `iterer` prend en paramètres - Un entier N, le nombre d'éléments de notre liste. - Un entier q pour initialiser la liste. - Un entier `niter`, le nombre d'itérations à effectuer.

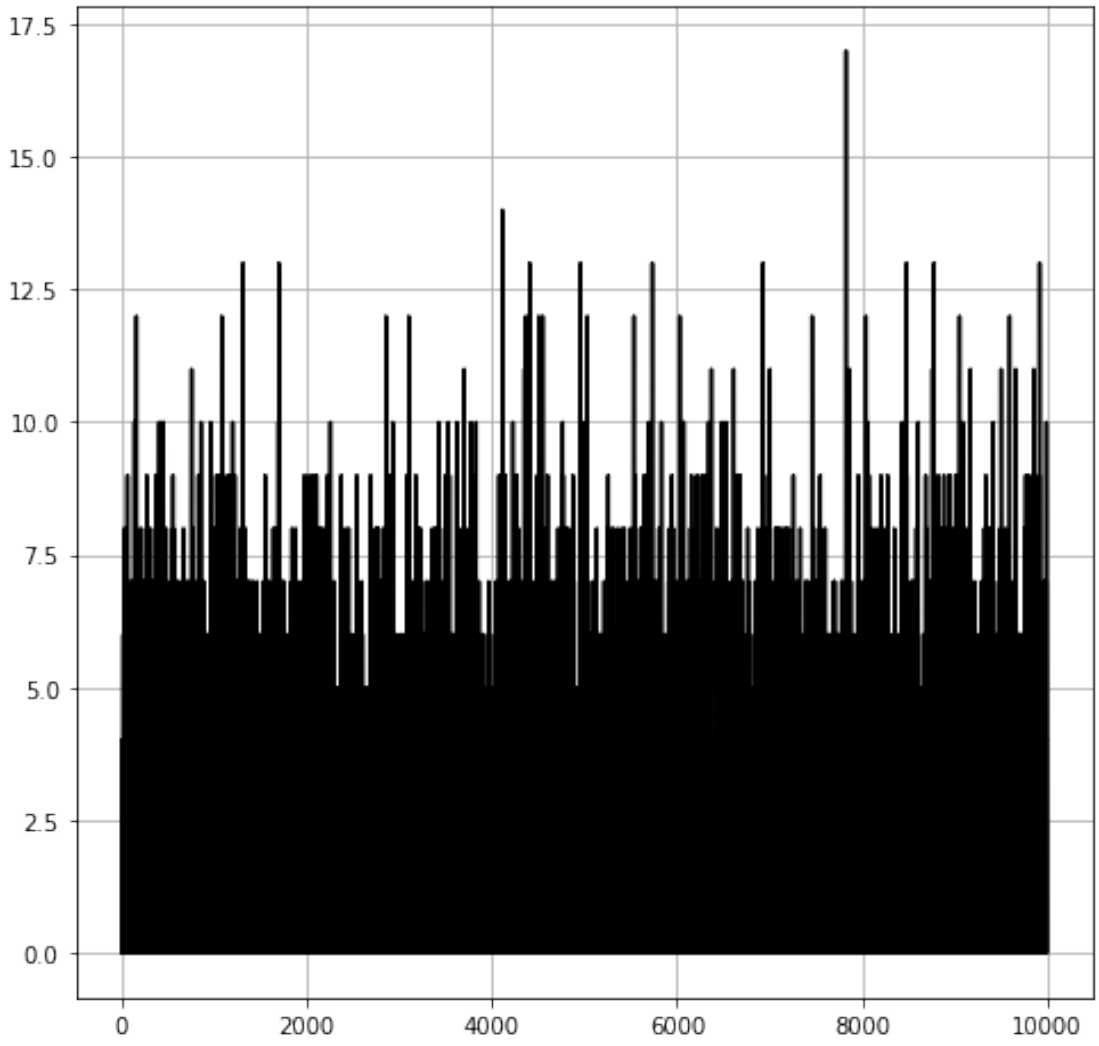
La fonction renvoie la liste S après itérations.

```
In [3]: def iterer(N, q, niter):
            S = [q for i in range(N)]
            for k in range(niter):
                    i = random.randint(0, N - 1)
                    if S[i] != 0:
                        j = random.randint(0, N - 1)
```

```
                S[j] += 1
                S[i] -= 1
        return S
```

Voici un exemple.

```
In [4]: N = 10000
        S = iterer(N, 2, 100000)
        plt.plot(S, 'k')
        plt.grid()
```



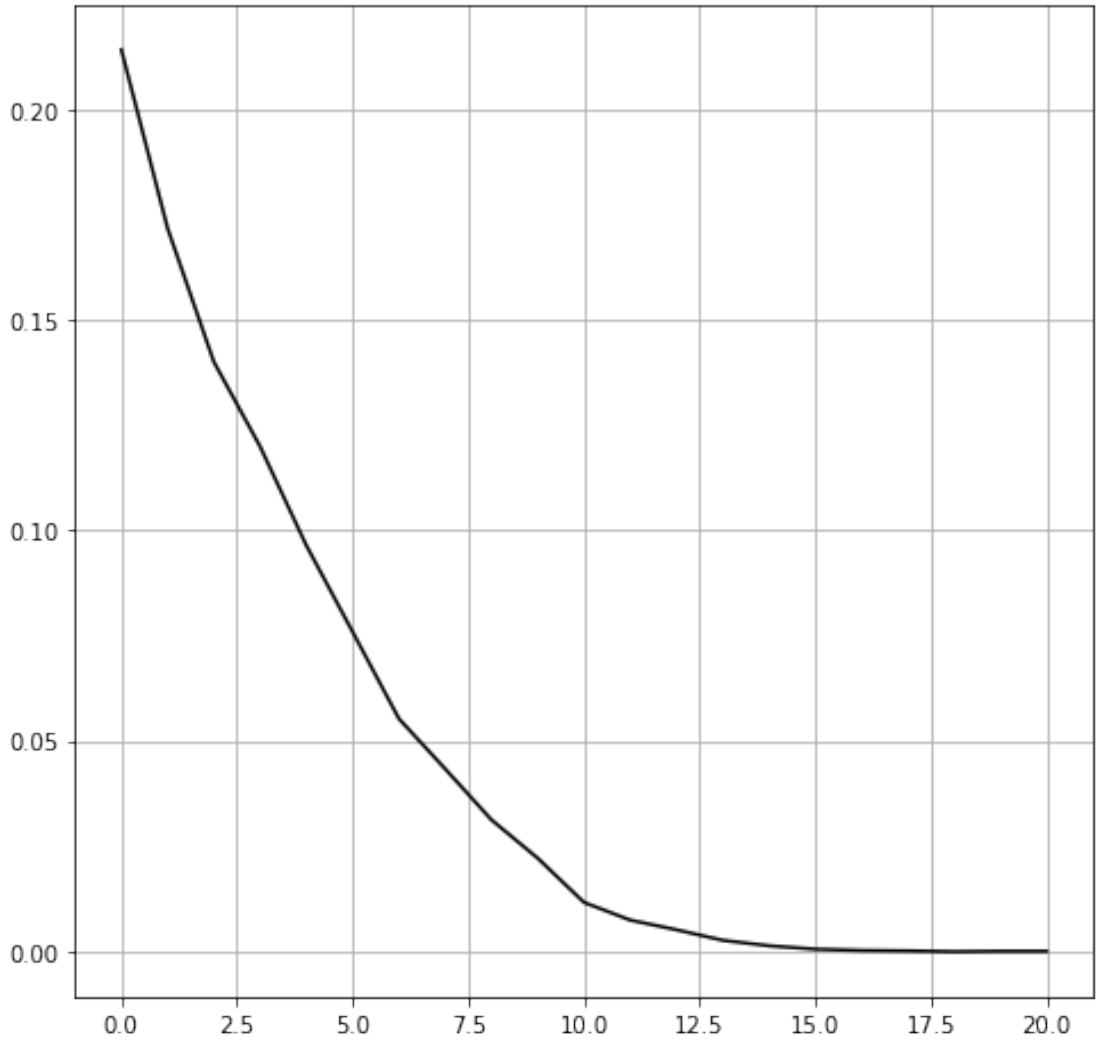
Ce que nous voyons ci-dessus c'est la liste S après un certain nombre d'itérations. C'est assez confus, alors intéressons-nous à **l'histogramme** S.

1.1.3 1.3 L'histogramme

La fonction `histogramme` renvoie ... **l'histogramme** de la liste S. En clair, elle renvoie une liste h telle que $h[0]$ est la probabilité qu'un élément de S ait la valeur 0, $h[1]$ est la probabilité qu'un élément de S ait la valeur 1, etc. En fait de probabilités, je devrais parler de **fréquences**. ne voulant pas entrer dans les détails pour ne pas lasser le lecteur, je serai assez léger sur la rigueur du vocabulaire tout au long du notebook.

```
In [5]: def histogramme(S):
            N = len(S)
            M = max(S)
            h = (M + 1) * [0]
            for i in range(N):
                    h[S[i]] += 1 / N
            return h
```

```
In [6]: S = iterer(10000, 3, 100000)
        h = histogramme(S)
        plt.plot(h, 'k')
        plt.grid()
```



On obtient pour l'histogramme une fonction décroissante. Est-ce toujours le cas ? Faites des tests, changez les valeurs des paramètres, vous verrez qu'effectivement, si `niter` est "assez grand" on retrouve des courbes similaires.

Quelques statistiques ? La fonction `moment` prend un histogramme h et un entier k en paramètres. Elle renvoie le **moment d'ordre** k de h, c'est à dire le réel

$$\mu(h,k) = \sum_{i=0}^{n-1} i^k h_i$$

où n est la taille de la liste h.

```
In [7]: def moment(h, k):
            s = 0
            n = len(h)
            for i in range(n):
```

```
            s += i ** k * h[i]
        return s
```

Pour $k = 0$, cela doit faire 1 à tout prix. Pour $k = 1$, on obtient la **moyenne** (ou, plus pompeusement, **l'espérance mathématique**) de h.

Il est aussi facile de définir variance et écart-type de h :

$$V(h) = \mu(h,2) - \mu(h,1)^2$$

$$\sigma(h) = \sqrt{V(h)}$$

```
In [8]: def moyenne(h): return moment(h, 1)

        def variance(h):
            return moment(h, 2) - moyenne(h) ** 2

        def ecart_type(h):
            return math.sqrt(variance(h))
```

Voici ce que cela donne pour notre jeu.

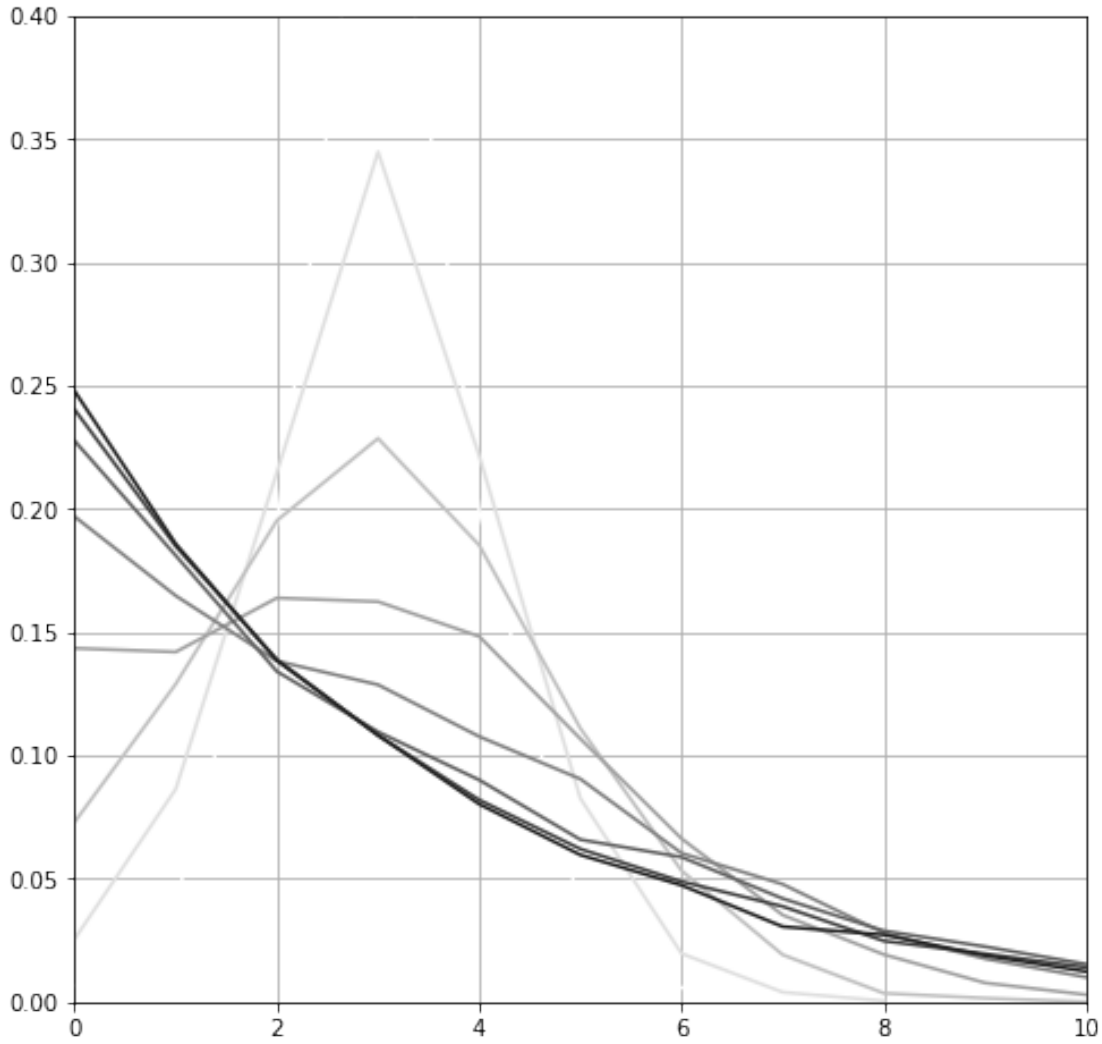
```
In [9]: S = iterer(10000, 2, 100000)
        h = histogramme(S)
        print(moyenne(h))
        print(ecart_type(h))
```

```
2.0000000000000036
2.2058558429779627
```

Une moyenne de 2 ne devrait pas vous surprendre ...

Tentons une petite expérience. Traçons les histogrammes obtenus pour différentes valeurs de `niter` sur un même graphique. Les courbes claires correspondent à de petites valeurs de `niter`, les courbes sombres à de grandes valeurs.

```
In [10]: ks = [2 ** k for k in range(12, 20)]
         n = len(ks)
         plt.xlim(0, 10)
         plt.ylim(0, 0.4)
         for k in range(n):
             S = iterer(10000, 3, ks[k])
             h = histogramme(S)
             c = 1. - k / n
             plt.plot(h, color=(c, c, c))
         plt.grid()
```



Les histogrammes ont l'air de "tendre" vers une sorte d'histogramme limite : le système $\mathcal{S}$ que nous essayons de modéliser tend-il vers un état limite ? Mais qu'est-ce qu'un **état** ? Nous allons voir qu'il existe deux sortes d'états : les micro-états et les macro-états.

1.2 2. Un peu de théorie

1.2.1 2.1 Micro-états, macro-états

2.1.1 Micro-états Reprenons notre jeu favori. La liste S, de taille N, est un système possédant une énergie égale à $E = qNE_0$, où $E_0 > 0$ est l'énergie d'un "quantum". Cette énergie est répartie entre les différentes cases de la liste. Combien y a-t-il de configurations possibles de la liste S ? Un nombre gigantesque. Vraiment énorme. Nous appellerons chacune de ces configurations un **micro-état** du système. Une façon fun de compter les micro-états est la suivante :

On dispose de $Nq + N$ objets, dont N sont des bâtons et Nq sont des étoiles. Les étoiles représentent les quanta d'énergie, les bâtons sont les bords gauches des cases de la liste S. Histoire de me faire bien comprendre, regardons l'exemple ci-dessous. Prenons $N = 4$ et $q = 3$. Le schéma

$$|*|****|***|****$$

représente la liste $S = [1, 4, 3, 4]$.

Combien y a-t-il de façon de placer les bâtons parmi les étoiles ? Le premier bâton se trouve évidemment toujours au début, c'est le bord gauche de la zéroième case de S. Il nous reste à placer $N-1$ bâtons, c'est à dire à choisir $N-1$ objets parmi les $Nq+N-1$ restants et les décréter "bâtons". Les autres sont des étoiles. Au total, nous avons donc

$$\Omega = \binom{Nq+N-1}{N-1} = \frac{(Nq+N-1)!}{(N-1)!(Nq)!} \quad \text{micro} - \text{tats}$$

Faisons intervenir $E = NqE_0$:

$$\Omega = \frac{(E/E_0+N-1)!}{(N-1)!(E/E_0)!} = \frac{1}{(N-1)!}(\frac{E}{E_0}+1)(\frac{E}{E_0}+2)\dots(\frac{E}{E_0}+N-1)$$

Histoire d'avoir une idée de l'ordre de grandeur du nombre de micro-états de notre jeu, voici une fonction `log_Omega` qui renvoie $\log\Omega$ (en base 10). En effet, il est inutile de vouloir calculer Ω, c'est un nombre bien trop gigantesque pour Python.

```
In [11]: def log_Omega(N, q):
             s = 0
             for k in range(1, N):
                 s += math.log(N * q + k, 10) - math.log(k, 10)
             return s
```

Un petit exemple ?

```
In [12]: log_Omega(10000, 3)
```

```
Out[12]: 9765.823327050703
```

Pour $N = 10000$ et $q = 3$ nous avons environ 10^{9800} micro-états, un 1 avec 9800 zéros derrière ... c'est beaucoup, le nombre d'atomes dans l'univers est, en étant TRÈS généreux, 10^{100}.

2.1.2 Macro-états Lorsque nous observons l'histogramme h, nous ne retenons que certaines informations associées à la liste. Combien de cases de valeur nulle ? De valeur 1 ? etc. Deux listes peuvent avoir le même histogramme et être totalement différentes. Nous appellerons l'histogramme un **macro-état**. Un macro-état pour notre jeu est donc une suite $(h_j)_{j\geq 0}$ de multiples entiers de $\frac{1}{N}$ telle que

$$\sum_{j=0}^{\infty} jh_j = q \quad \text{et} \quad \sum_{j=0}^{\infty} h_j = 1$$

Combien notre jeu possède-t-il de macro-états ? Moins que de micro-états, évidemment, puisque chaque macro-état correspond à plusieurs (beaucoup de) micro-états. Le dénombrement des macro-états de notre système est une question ouverte, si quelqu'un a une idée je suis preneur.

1.2.2 2.2 Hypothèses de travail

Soit $\mathcal{S}$ un système. Nous supposons que le système peut être décrit par un très grand nombre de micro-états ayant tous la même probabilité. Nous pouvons par exemple penser à un système formé de particules ayant chacune une position, une vitesse, etc. Un micro-état serait la donnée de tous les paramètres associés à une particule, ceci pour toutes les particules.

Ce que le physicien mesure, en revanche, ce sont des quantités issues des macro-états du système (la pression d'un gaz par exemple). Tous les macro-états n'ont pas la même probabilité, mais nous allons faire l'hypothèse suivante :

Hypothèse : Parmi tous les macro-états possibles, un système "choisit" celui qui correspond au plus grand nombre possible de micro-états.

On peut montrer que cette hypothèse est une conséquence des trois hypothèses ci-dessous :

1. Tous les micro-états du système ont la même probabilité.
2. La dynamique interne du système est telle que les micro-états du système changent continuellement.
3. Si on attend suffisamment longtemps (cela peut être très, très, très longtemps :-), le système explorera tous les micro-états possibles et passera un temps égal dans chacun d'entre-eux.

Munis de cette magnifique hypothèse, nous allons définir le mot **température**.

1.2.3 2.3 Température

Considérons deux systèmes qui peuvent échanger de l'énergie entre-eux, mais avec rien d'autre. Le système $\mathcal{S}_1$ a une énergie E_1 et le système $\mathcal{S}_2$ a une énergie E_2. L'énergie totale

$$E = E_1 + E_2$$

est donc, avec nos hypothèses, constante. Les deux systèmes $\mathcal{S}_1$ et $\mathcal{S}_2$ procèdent à des échanges d'énergie, et au bout d'un certain temps un équilibre s'établit : **l'équilibre thermique**. Cet équilibre est celui qui maximise le nombre de micro-états.

Le système $\mathcal{S}_i$ ($i = 1, 2$) peut être dans n'importe lequel de $\Omega_i(E_i)$ micro-états. Un micro-état de notre système global est donc un couple formé d'un micro-état de $\mathcal{S}_1$ et d'un micro-état de $\mathcal{S}_2$. Le nombre total de micro-états du système est ainsi

$$\Omega_1(E_1)\Omega_2(E_2)$$

Nous cherchons donc à maximiser cette quantité. Nous allons travailler dans le cadre où Ω_i est une fonction continue de E_i. Mais comment une fonction à valeurs entières peut-elle être continue ? Si cela vous chagrine, rappelez-vous le nombre de micro-états de notre jeu. Et travaillez sur la fonction $10^{-9800}\Omega_i$ qui est un nombre entre 0 et 1, avec 9800 chiffres après la virgule. Du coup, l'hypothèse de quantités continues n'est plus du tout absurde :-). Dérivons par rapport à E_1. En un maximum cette dérivée doit être nulle.

$$\frac{d}{dE_1}(\Omega_1(E_1)\Omega_2(E_2)) = 0$$

Développons :

$$\Omega_2(E_2)\frac{d}{dE_1}\Omega_1(E_1) + \Omega_1(E_1)\frac{d}{dE_2}\Omega_2(E_2)\frac{dE_2}{dE_1} = 0$$

Comme $E_1 + E_2 = E$ est constante, on a $\frac{dE_2}{dE_1} = -1$. Ainsi,

$$\Omega_2(E_2)\frac{d}{dE_1}\Omega_1(E_1) = \Omega_1(E_1)\frac{d}{dE_2}\Omega_2(E_2)$$

ce que l'on peut encore écrire

$$\frac{1}{\Omega_1(E_1)}\frac{d}{dE_1}\Omega_1(E_1) = \frac{1}{\Omega_2(E_2)}\frac{d}{dE_2}\Omega_2(E_2)$$

ou encore

$$\frac{d}{dE_1}\ln\Omega_1(E_1) = \frac{d}{dE_2}\ln\Omega_2(E_2)$$

Posons

$$\frac{1}{k_B T_i} = \frac{d\ln\Omega_i(E_i)}{dE_i}$$

où $k_b = 1.3807 \times 10^{-23} JK^{-1}$ est la **constante de Boltzmann**. Le nombre T_i est la **température** du système $\mathcal{S}_i$, de sorte que l'équilibre thermique s'écrit

$$T_1 = T_2$$

Ceci nous suggère la

Définition : La température T d'un système $\mathcal{S}$ est donnée par

$$\frac{1}{k_B T} = \frac{d\ln\Omega(E)}{dE}$$

où Ω est le nombre de micro-états du système, vu comme une fonction de son énergie E.

Remarque : Si l'apparition du nombre magique k_B vous inquiète, il vous suffit de remarquer que la valeur de la constante en question dépend des unités choisies. Historiquement, la différence entre la température d'ébullition de l'eau et celle de sa fusion a été fixée par un certain M. Celsius à 100. Pourquoi pas ? Puis un certain M. Kelvin a décidé que l'unité de température serait une simple translation de l'échelle Celsius. Si M. Celsius avait décidé de mettre 6.022×10^{23} à la place de 100, k_B aurait évidemment une tout autre valeur.

Exercice : quelle valeur ?

1.2.4 2.4 Probabilités

Considérons à nouveau deux systèmes que nous appellerons $\mathcal{R}$ et $\mathcal{S}$. Nous faisons les deux hypothèses suivantes :

- Le système $\mathcal{R}$ est "énorme", nous l'appellerons le **réservoir**. On peut retirer ou ajouter à ce système de l'énergie sans que sa température soit modifiée de façon appréciable.
- Pour chaque énergie possible ε du système $\mathcal{S}$ il y a un unique micro-état de $\mathcal{S}$ associé. Avec des notations évidentes, $\Omega_{\mathcal{S}}(\epsilon) = 1$.

Imaginez par exemple que $\mathcal{S}$ est une casserole d'eau, plongée dans l'océan $\mathcal{R}$. Comme envisagé plus haut, les deux systèmes peuvent échanger de l'énergie et nous supposons que l'énergie

totale du système, E, est constante. À l'équilibre thermique les deux systèmes ont la même température qui est, avec nos hypothèses, celle du réservoir (de combien l'océan se réchauffe-t-il quand vous y versez une casserole d'eau chaude ?).

La probabilité $P(\epsilon)$ que le système $\mathcal{S}$ ait l'énergie ε est proportionnelle au nombre total de micro-états, qui est $\Omega(E) = \Omega_{\mathcal{R}}(E-\epsilon) \times \Omega_{\mathcal{S}}(\epsilon)$:

$$P(\epsilon) = \lambda \Omega_{\mathcal{R}}(E-\epsilon)$$

où $\lambda > 0$ est choisi pour que P soit une probabilité (la somme sur toutes les valeurs possibles de ϵ doit valoir 1). On a, en faisant un développement limité à l'ordre 1,

$$\ln \Omega_{\mathcal{R}}(E-\epsilon) = \ln \Omega_{\mathcal{R}}(E) - \epsilon \frac{d \ln \Omega_{\mathcal{R}}(E)}{dE} + o(\epsilon)$$

ou encore

$$\ln \Omega_{\mathcal{R}}(E-\epsilon) = \ln \Omega_{\mathcal{R}}(E) - \frac{\epsilon}{k_B T} + o(\epsilon)$$

d'où

$$\Omega_{\mathcal{R}}(E-\epsilon) = \Omega_{\mathcal{R}}(E) e^{-\epsilon/k_B T}(1 + o(\epsilon))$$

Admettons que dans le cadre où nous nous sommes placés on peut négliger le terme $o(\varepsilon)$. On obtient donc

$$P(\epsilon) = \lambda e^{-\epsilon/k_B T}$$

Pour chaque valeur de T (fixée par le réservoir), nous avons une distribution de probabilité sur l'ensemble des énergies du système $\mathcal{S}$. Elle est appelée la **distribution de Boltzmann**.

Après ce long intermède théorique, voyons ce que cela donne pour notre jeu !

1.3 3. Retour au jeu

Quelle est la valeur de la case $S[0]$? Eh bien c'est un entier naturel ... Mais encore ? Quelle est, disons, la probabilité que $S[0] = k$, où $k \in \mathbb{N}$? Regardons notre jeu comme la réunion de deux systèmes $\mathcal{R}$ et $\mathcal{S}$, où $\mathcal{S}$ est le système formé par la case $S[0]$ et $\mathcal{R}$, le réservoir, est la liste formée de toutes les autres cases. Nous pouvons maintenant appliquer ce que nous avons raconté dans la section précédente. Il faudrait en toute rigueur remplacer N par $N-1$ dans les calculs qui vont suivre (le réservoir n'a que $N-1$ cases), mais je fais confiance au lecteur pour vérifier que cela ne change pas les résultats obtenus : nous allons de toute façon faire tendre N vers l'infini.

1.3.1 3.1 Estimations

Souvenons-nous que dans notre jeu le nombre de micro-états est

$$\Omega = \frac{1}{(N-1)!}(\frac{E}{E_0}+1)(\frac{E}{E_0}+2)\dots(\frac{E}{E_0}+N-1)$$

On a donc

$$\ln \Omega = \sum_{k=1}^{N-1} \ln(\frac{E}{E_0}+k) - \ln(N-1)!$$

De là,

$$\frac{1}{k_B T} = \frac{d \ln \Omega}{dE} = \frac{1}{E_0} \sum_{k=1}^{N-1} \frac{1}{\frac{E}{E_0} + k} = \frac{1}{E_0} \sum_{k=1}^{N-1} \frac{1}{Nq + k}$$

Réécrivons tout cela sous la forme

$$\frac{E_0}{k_B T} = \frac{1}{Nq} \sum_{k=1}^{N-1} \frac{1}{1 + \frac{k}{Nq}}$$

Nos yeux experts voient apparaître une somme de Riemann associée à l'intégrale

$$\int_0^{\frac{1}{q}} \frac{dx}{1+x} = \ln(1 + \frac{1}{q})$$

Ainsi, en passant à la limite lorsque $N \to \infty$,

$$P(\epsilon) = \frac{\lambda}{E_0} e^{-\epsilon \ln(1+1/q)/E_0} = \frac{\lambda}{E_0} \frac{1}{(1 + \frac{1}{q})^{\epsilon/E_0}}$$

Que vaut λ ? Eh bien les énergies possibles sont les multiples entiers de E_0. Nous avons donc $\sum_{i=0}^{\infty} P(iE_0) = 1$. Or,

$$\sum_{i=0}^{\infty} \frac{1}{(1 + \frac{1}{q})^i} = \frac{1}{1 - \frac{1}{1+\frac{1}{q}}} = q + 1$$

Ainsi, $\lambda = \frac{E_0}{q+1}$:

$$P(\epsilon) = \frac{1}{q+1} \frac{1}{(1 + \frac{1}{q})^{\epsilon/E_0}}$$

ou encore, pour s'en tenir aux valeurs possibles de l'énergie :

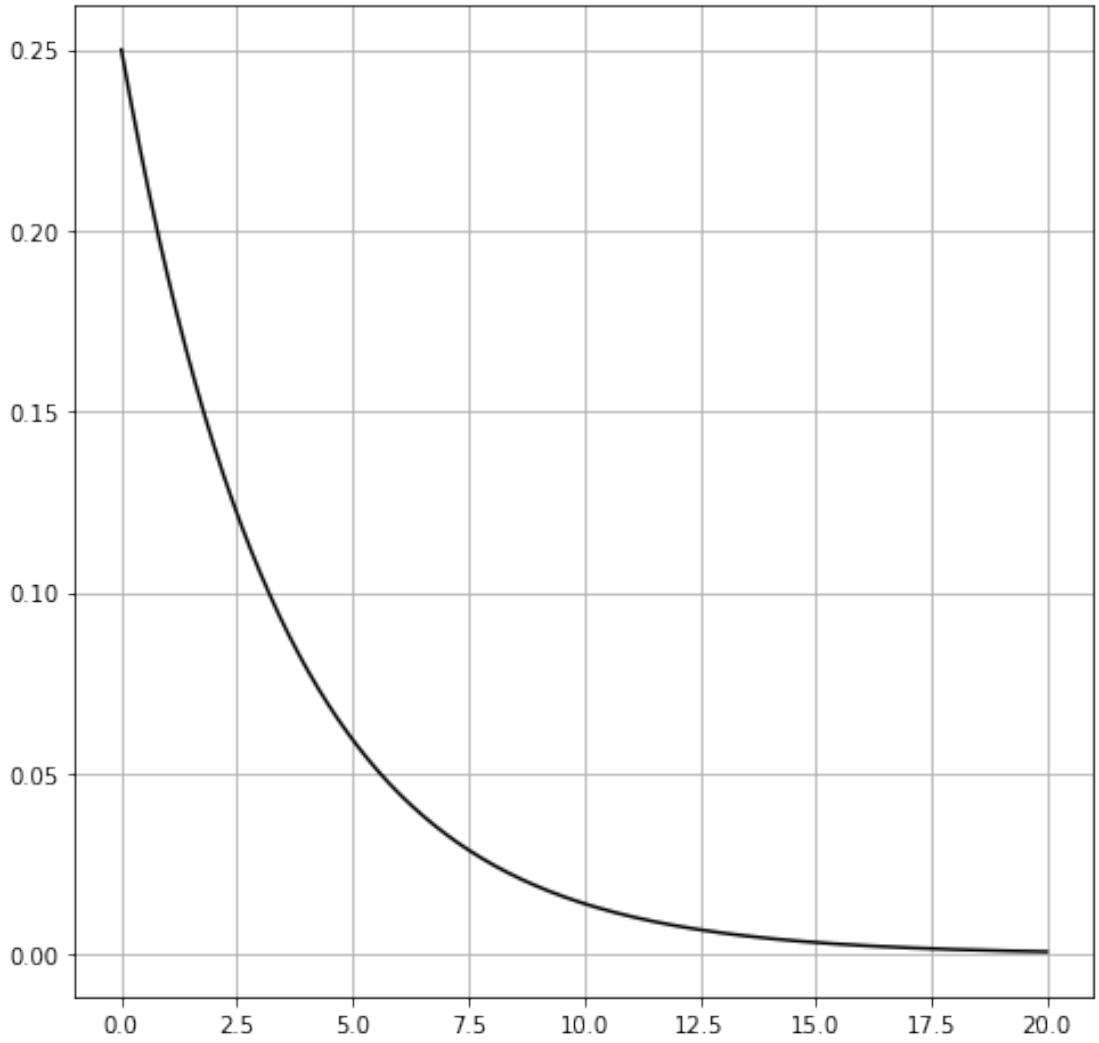
$$P(kE_0) = \frac{1}{q+1} \frac{1}{(1 + \frac{1}{q})^k}$$

La fonction `proba` ci-dessous prend deux paramètres k et q. Elle renvoie $P(kE_0)$, le paramètre q étant le paramètre fixé pour notre système.

```
In [13]: def proba(k, q):
             return 1 / (1 + 1 / q) ** k * 1 / (q + 1)
```

Voilà donc l'histogramme "limite" dont nous avons parlé plus haut, que nous obtiendrions après une infinité d'itérations de notre jeu.

```
In [14]: ks = [k / 100 for k in range(2000)]
         ys = [proba(k, 3) for k in ks]
         plt.plot(ks, ys, 'k')
         plt.grid()
```



Sur cet exemple, une case de la liste a une probabilité d'environ $\frac{1}{4}$ de contenir la valeur 0.

1.3.2 3.2 Intermède rigolo - Quelle est la température de notre jeu ?

Quelle est la température de notre jeu ? Les calculs précédents montrent que

$$T = \frac{E_0}{k_B \ln(1 + \frac{1}{q})}$$

Il va nous falloir prendre une décision quant à la valeur de E_0. Quelle est la masse d'une molécule d'azote ? La masse molaire du diazote est environ 28 grammes par mole.

```
In [15]: N_Avo = 6.022e23
         m_azote = 1 / N_Avo * 28 * 1e-3
         print(m_azote)
```

```
4.649618067087346e-26
```

Sachant que la vitesse d'une telle molécule dans l'air est de l'ordre de $500ms^{-1}$, son énergie cinétique est donc, en joules,

```
In [16]: vmoy = 500
         E0 = 0.5 * m_azote * vmoy ** 2
         print(E0)
```

```
5.812022583859183e-21
```

Prenons cette valeur pour E_0.

```
In [17]: def temp(q):
             kb = 1.3807e-23
             return E0 / kb / math.log(1 + 1 / q)
```

Tentons avec $q = 1$. N'attendons aucun miracle.

```
In [18]: temp(1) - 273.15
```

```
Out[18]: 334.1489178871808
```

Houla, 334 degrés Celsius, c'est chaud :-) ! Mais ce n'est pas 10^{-50} ou 10^{312}, on trouve un nombre qui ressemble vraiment à une température "normale".

J'aurais adoré trouver 25 degrés Celsius, la température de l'air par un beau jour du joli mois de mai, mais avec notre modèle ultra-simpliste c'était peine perdue ... Si vous voulez vraiment épater le monde scientifique, changez les limitations de vitesse pour les molécules et prenez une vitesse de $350ms^{-1}$:-).

```
In [19]: vmoy = 350
         E0 = 0.5 * m_azote * vmoy ** 2
         temp(1) - 273.15
```

```
Out[19]: 24.426469764718547
```

NB : Eh oui, 25^oC. Mais c'est pas beau de tricher :-).

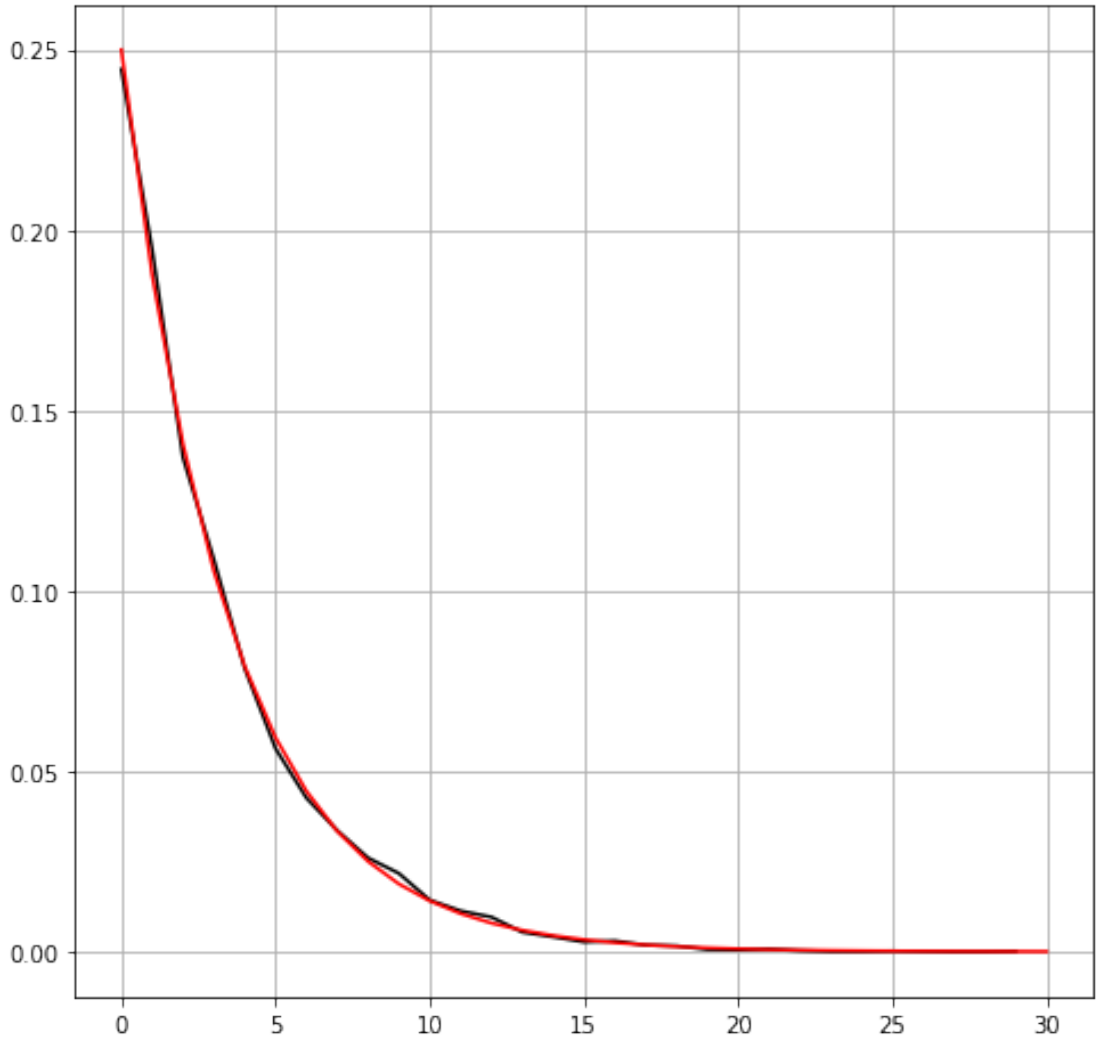
1.3.3 3.3 Le moment de vérité

Nous y voilà. Le moment fatidique c'est celui où l'on compare théorie et pratique. Affichons sur un même graphique l'histogramme de la liste S et la courbe de $P(\epsilon)$.

```
In [20]: def stats3(S, q, N):
             h = histogramme(S)
             plt.plot(h, color='k')
             xs = [proba(k, q) for k in range(len(h) + 1)]
             plt.plot(xs, 'r')
             plt.grid()
```

Allons-y. - En noir, l'histogramme de notre jeu. - En rouge, les probabilités prévues par la théorie.

```
In [21]: q = 3
         N = 10000
         S = iterer(N, q, 500000)
         stats3(S, q, N)
```



Nous sommes satisfaits au delà de tout ce que nous avions pu espérer :-). Et encore, j'ai pris un nombre d'itérations pas trop grand. Parce que si vous augmentez le nombre d'itérations, vous ne verrez plus du tout la courbe noire (qui a été tracée en premier par `matplotlib`).

```
In [ ]:
```