Loi normale et bruit gaussien

Comment le hasard structuré aide une IA à démarrer et à créer

Hors-programme — à lire avant de commencer

Vérification faite auprès du texte officiel du programme (BO spécial n°8 du 25 juillet 2019, programme actuellement en vigueur) : la loi normale ne figure pas dans le programme de spécialité mathématiques de terminale. La partie « Probabilités » de ce programme se limite au schéma de Bernoulli, à la loi binomiale, aux sommes de variables aléatoires et à la loi des grands nombres.

Ce document est donc un enrichissement hors-programme, à réserver aux élèves curieux, à l'option mathématiques expertes, ou à une ouverture vers l'enseignement supérieur (la loi normale reste incontournable en BTS, classes préparatoires et études scientifiques). Contrairement à une simple analogie mathématique inventée pour l'occasion, les deux mécanismes présentés ici sont authentiquement utilisés dans l'IA moderne : l'initialisation aléatoire des poids d'un réseau de neurones, et les modèles de diffusion qui génèrent des images.

Niveau : Hors-programme (enrichissement) — bases de probabilités de Première/Terminale requises

Introduction : pourquoi une IA a-t-elle besoin de hasard ?

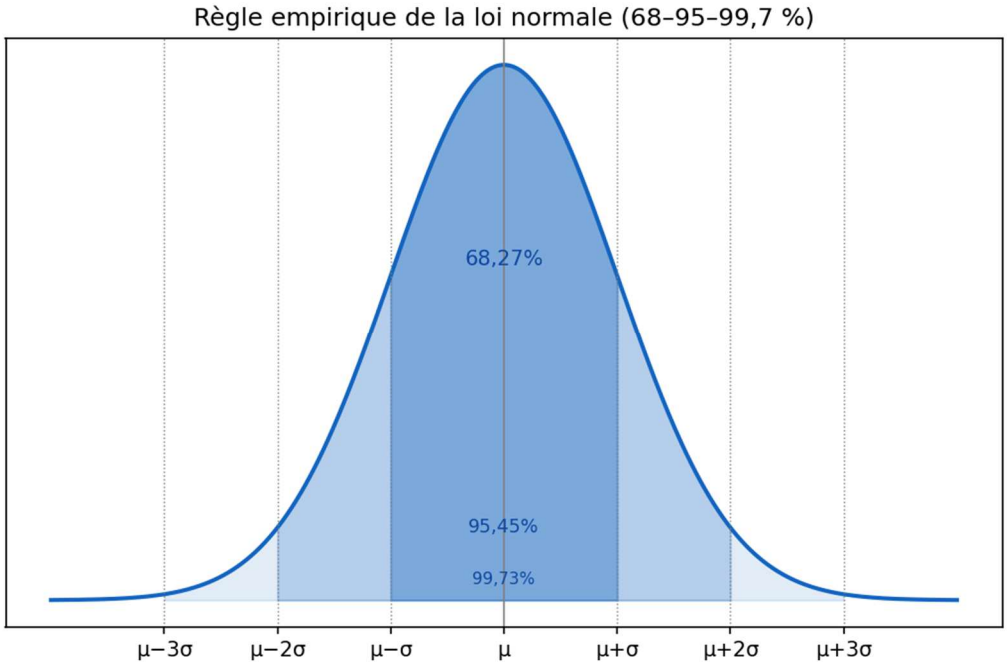
On pourrait penser qu'une IA, qui manipule des calculs précis (descente de gradient, dérivées, probabilités), n'a aucun usage pour le hasard « pur ». C'est faux : deux étapes essentielles reposent sur un tirage aléatoire selon une loi normale.

- **Avant l'entraînement :** les poids d'un réseau de neurones doivent être initialisés à des valeurs de départ, tirées aléatoirement selon une loi normale.
- **Pour générer une image :** les modèles de génération d'images (comme ceux utilisés pour créer des illustrations) partent d'un bruit purement aléatoire gaussien, qu'ils transforment progressivement en image.

Partie A — Rappel : la loi normale et sa règle empirique

Niveau : Hors-programme (enrichissement)

On rappelle qu'une variable aléatoire X suit une loi normale de paramètres μ (moyenne) et σ (écart-type), notée $X \rightsquigarrow \mathcal{N}(\mu ; \sigma^2)$, si sa densité de probabilité a la forme d'une courbe en cloche centrée en μ.



Règle empirique (dite « règle 68–95–99,7 »)

$P(\mu-\sigma < X < \mu+\sigma) \approx 0{,}6827$

$P(\mu-2\sigma < X < \mu+2\sigma) \approx 0{,}9545$

$P(\mu-3\sigma < X < \mu+3\sigma) \approx 0{,}9973$

1. On considère $X \rightsquigarrow \mathcal{N}(0\ ;\ 0{,}01)$ (donc $\sigma=0{,}1$). Donne, sans calculatrice, une valeur approchée de $P(-0{,}1 < X < 0{,}1)$, puis de $P(-0{,}2 < X < 0{,}2)$.
2. Toujours pour $X \rightsquigarrow \mathcal{N}(0\ ;\ 0{,}01)$, donne une valeur approchée de $P(X > 0{,}3)$ (tu pourras utiliser la symétrie de la courbe autour de 0).
3. Ces probabilités dépendent-elles de la valeur de σ, ou seulement du nombre d'écarts-types considérés ? Justifie à partir de la règle empirique.

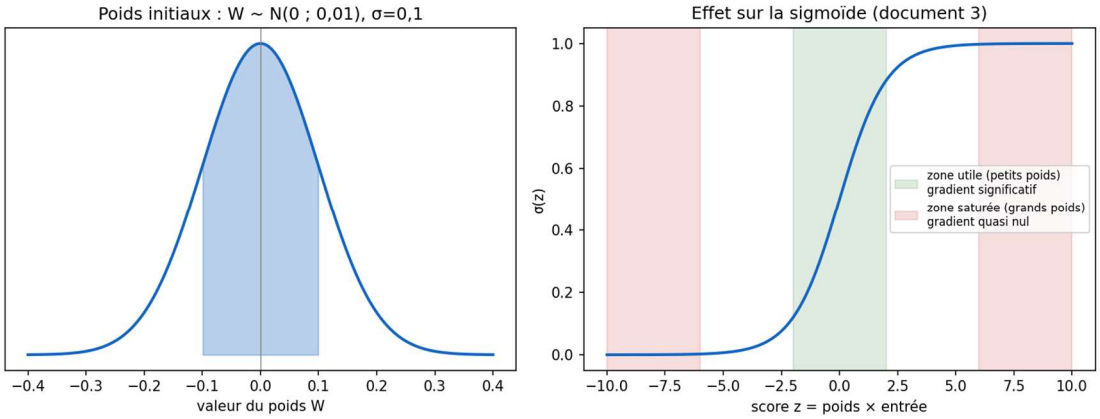
Partie B — Initialiser les poids d'un réseau de neurones

Niveau : Hors-programme (enrichissement)

Avant de commencer la descente de gradient (document 0), un réseau de neurones doit partir d'une valeur initiale pour chacun de ses poids. On pourrait penser que 0 est un bon point de départ neutre — c'est en réalité un très mauvais choix.

4. Si tous les poids d'une même couche sont initialisés à 0, tous les neurones de cette couche calculent exactement la même chose, et reçoivent donc exactement le même gradient à chaque étape de la descente de gradient (document 0). Que va-t-il se passer au fil des itérations ? Ces neurones vont-ils un jour se différencier les uns des autres ?

Pour éviter ce problème (appelé rupture de symétrie), on initialise chaque poids par un tirage aléatoire indépendant selon une loi normale centrée en 0 : $W \rightsquigarrow \mathcal{N}(0\ ;\ \sigma^2)$, avec un σ soigneusement choisi — ni trop petit, ni trop grand.



5. On choisit $\sigma = 0{,}1$. En utilisant la partie A, donne une valeur approchée de $P(-0{,}1 < W < 0{,}1)$ et de $P(|W| > 0{,}3)$ pour un poids W tiré selon cette loi.
6. Rappelle (document 3) la formule de la dérivée de la fonction sigmoïde : $\sigma'(z) = \sigma(z)(1-\sigma(z))$. D'après le graphique ci-dessus, que vaut approximativement $\sigma'(z)$ lorsque $|z|$ est grand (par exemple $|z| > 6$) ?
7. En déduire pourquoi initialiser les poids avec un σ beaucoup trop grand (par exemple $\sigma=10$) risquerait de bloquer l'apprentissage dès le départ, à cause du score $z = \text{poids} \times \text{entrée}$ obtenu.
8. Pourquoi un σ beaucoup trop petit (par exemple $\sigma=0{,}0001$) poserait-il, à l'inverse, un problème pour la rupture de symétrie évoquée en question 1 ?

Partie C — Bruiter puis débruiter : les modèles de diffusion

Niveau : Hors-programme (enrichissement)

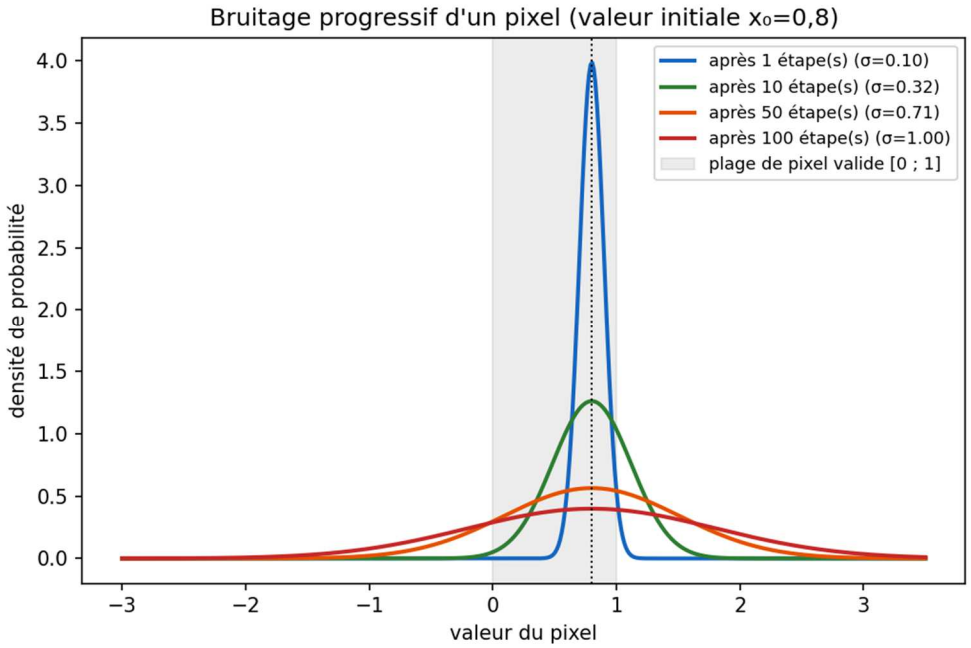
Certains modèles d'IA génèrent des images en partant de bruit pur, puis en le transformant progressivement en image cohérente. Pour comprendre ce principe, on l'étudie « à l'envers » : on part d'une image, et on lui ajoute du bruit gaussien, étape par étape, jusqu'à obtenir du bruit pur (ce qu'on appelle le processus direct de diffusion).

On simplifie à l'extrême en ne suivant qu'un seul pixel, de valeur initiale $x_0 = 0{,}8$ (sur une échelle de 0 à 1). À chaque étape, on ajoute un bruit indépendant suivant $\mathcal{N}(0\ ;\ 0{,}01)$ ($\sigma=0{,}1$ par étape) :

Modèle de bruitage

$\mathbf{x_n = x_0 + \varepsilon_1 + \varepsilon_2 + \ldots + \varepsilon_n}$, **avec chaque $\varepsilon_i \rightsquigarrow \mathcal{N}(0\ ;\ 0{,}01)$ indépendant**

On admet un résultat de cours (hors programme, donné ici) : la somme de n bruits indépendants de variance 0,01 suit une loi normale de variance $0{,}01 \times n$. La variable x_n suit donc $\mathcal{N}(x_0\ ;\ 0{,}01n)$, c'est-à-dire un écart-type égal à $0{,}1 \times \sqrt{n}$.



9. Calcule l'écart-type de x_{100} (après 100 étapes de bruitage).
10. En déduire, à l'aide de la calculatrice (loi normale $\mathcal{N}(0{,}8\ ;\ 1)$), une valeur approchée de $P(0 < x_{100} < 1)$, c'est-à-dire la probabilité que le pixel bruité reste encore dans une plage de valeur « valide » (arrondis à 10^{-3} près).
11. Compare cette probabilité à ce qu'elle vaudrait juste après l'étape 1 (où $x_1 \rightsquigarrow \mathcal{N}(0{,}8\ ;\ 0{,}01)$ — tu peux utiliser la partie A). Que peux-tu en conclure sur la quantité d'information encore présente sur la valeur d'origine x_0, après 100 étapes de bruitage ?

Un modèle de diffusion apprend, par descente de gradient (document 0), à faire le chemin inverse : deviner puis retirer le bruit ajouté à chaque étape, en minimisant une fonction de coût qui n'est autre que l'erreur quadratique moyenne (document 4) entre le bruit réellement ajouté et le bruit prédit par le modèle. En répétant ce « débruitage » depuis un bruit pur, le modèle finit par produire une image entièrement nouvelle et cohérente.

Ce que ce modèle simplifie, et ce que fait réellement une IA comme Claude

Ce que ce modèle simplifie	Ce que fait réellement un modèle comme Claude
Un seul pixel, avec un bruit ajouté de variance constante à chaque étape.	Des millions de pixels (ou de valeurs, pour un modèle de texte) bruités simultanément, avec un niveau de bruit qui suit un « planning » soigneusement conçu (plus faible au début, plus fort à la fin du bruitage).
Un choix de σ pour l'initialisation des poids présenté comme un simple compromis « ni trop grand ni trop petit ».	Des formules précises (initialisations dites de Xavier ou de He), qui calculent σ en fonction du nombre de neurones d'entrée et de sortie de chaque couche, pour garantir mathématiquement que les signaux ne s'éteignent ni n'explosent en traversant un réseau très profond.
Un processus de débruitage décrit en une phrase.	Un immense réseau de neurones (souvent une architecture appelée U-Net ou un transformeur) entraîné sur des milliards d'images, qui apprend à prédire le bruit à chaque étape parmi des centaines d'étapes de débruitage.

Le principe central — un tirage aléatoire structuré par une loi normale, puis une descente de gradient qui apprend à l'exploiter ou à l'inverser — reste, lui, exactement ce qui se passe dans les IA génératrices d'images actuelles.

Exercices d'entraînement

Exercice 1 — Un autre écart-type d'initialisation

Niveau : Hors-programme (enrichissement)

On initialise les poids d'une couche avec $W \rightsquigarrow \mathcal{N}(0\ ;\ 0{,}0025)$ (donc $\sigma = 0{,}05$).

12. Donne une valeur approchée de $P(-0{,}05 < W < 0{,}05)$.
13. Donne une valeur approchée de $P(|W| > 0{,}15)$.
14. Compare ces résultats à ceux obtenus en partie B avec $\sigma=0{,}1$. Qu'en conclus-tu sur le rôle de σ dans la règle empirique ?

Exercice 2 — Un autre scénario de bruitage

Niveau : Hors-programme (enrichissement)

On bruite un pixel de valeur initiale $x_0=0{,}6$, avec un bruit de variance 0,04 à chaque étape ($\sigma=0{,}2$ par étape), pendant 25 étapes.

15. Calcule l'écart-type de x_{25}.
16. En déduire une valeur approchée de $P(0 < x_{25} < 1)$ (arrondis à 10^{-3} près).
17. Ce pixel a-t-il, après ces 25 étapes, une distribution plus ou moins étalée que celle de x_{100} calculée en partie C ? Est-ce cohérent avec les paramètres choisis (moins d'étapes, mais un bruit plus fort à chaque étape) ?

Exercice 3 — Rupture de symétrie (réflexion)

Niveau : Hors-programme (enrichissement)

Deux neurones A et B d'une même couche sont tous les deux initialisés avec le même poids $w_0 = 0$. On admet que, dans ce cas, les deux neurones reçoivent, à chaque itération de descente de gradient, exactement le même gradient.

18. Montre, à l'aide d'un raisonnement par récurrence informel, que les poids de A et de B resteront égaux après n'importe quel nombre d'itérations.
19. En quoi ce résultat est-il problématique pour un réseau censé comporter plusieurs neurones « spécialisés » dans une même couche ?

Corrigé

Partie A

A.1. D'après la règle empirique : $P(-0{,}1<X<0{,}1) \approx 0{,}6827$, et $P(-0{,}2<X<0{,}2) \approx 0{,}9545$.

A.2. $P(X>0{,}3)$ correspond à la moitié de la probabilité restante en dehors de $[-0{,}3\ ;\ 0{,}3]$: $P(X>0{,}3) = (1-0{,}9973)/2 \approx 0{,}00135$.

A.3. Ces probabilités ne dépendent que du nombre d'écarts-types (1σ, 2σ, 3σ), pas de la valeur précise de σ : la règle empirique s'applique de la même façon quelle que soit la valeur de σ, car elle porte sur des multiples de σ, pas sur des valeurs absolues.

Partie B

B.1. Si tous les poids partent de 0 et reçoivent exactement le même gradient à chaque étape, ils seront mis à jour de exactement la même façon à chaque itération : ils resteront donc égaux entre eux indéfiniment. Les neurones ne se différencieront jamais.

B.2. D'après la partie A ($\sigma=0{,}1$) : $P(-0{,}1<W<0{,}1) \approx 0{,}6827$. $P(|W|>0{,}3) = P(|W|>3\sigma) \approx 0{,}0027$ (les deux queues à 3σ, voir A.2 × 2).

B.3. D'après le graphique, lorsque $|z|$ est grand, $\sigma(z)$ est très proche de 0 ou de 1, donc $\sigma(z)\times(1-\sigma(z))$ est très proche de 0 : $\sigma'(z) \approx 0$.

B.4. Avec $\sigma=10$, la plupart des poids seraient très grands en valeur absolue, ce qui rendrait le score z = poids × entrée très grand ou très négatif pour beaucoup de neurones. D'après B.3, cela donnerait un gradient $\sigma'(z)$ quasiment nul : la descente de gradient (document 0) ne pourrait quasiment plus ajuster les poids, car $\partial L/\partial a=(\hat{y}-y)\times x$ dépend de la dérivée de la sigmoïde (document 3) qui serait ici presque nulle. L'apprentissage serait bloqué dès le départ (on parle de « saturation » ou de gradient qui s'évanouit).

B.5. Avec un σ extrêmement petit, tous les poids initiaux seraient presque tous égaux à 0 (très proches les uns des autres) : le problème de rupture de symétrie évoqué en B.1 réapparaîtrait presque à l'identique, car les neurones démarreraient avec des valeurs quasiment indiscernables.

Partie C

C.1. Écart-type de $x_{100} = 0{,}1\times\sqrt{100} = 0{,}1\times10 = 1$.

C.2. $x_{100} \rightsquigarrow \mathcal{N}(0{,}8\ ;\ 1)$. En centrant-réduisant : $P(0<x_{100}<1) = P((0-0{,}8)/1 < Z < (1-0{,}8)/1) = P(-0{,}8<Z<0{,}2) \approx 0{,}367$ (valeur obtenue à la calculatrice).

C.3. Juste après l'étape 1, $x_1 \rightsquigarrow \mathcal{N}(0{,}8\ ;\ 0{,}01)$ ($\sigma=0{,}1$) : $P(0<x_1<1)$ est en réalité très proche de 1 (l'intervalle [0;1] représente ici plus de 8 écarts-types autour de 0,8, donc une probabilité extrêmement proche de 1 d'après la règle empirique, bien au-delà de 3σ). Comparé à C.2 ($\approx$0,367 après 100 étapes), on constate que le bruit a fortement « dilué » l'information sur x_0 : après 100 étapes, on ne peut quasiment plus dire si le pixel bruité provient encore d'un pixel valide, alors qu'après une seule étape, l'information initiale est encore quasiment intacte.

Exercice 1

1. $\sigma=0{,}05$ correspond exactement à 1σ : $P(-0{,}05<W<0{,}05) \approx 0{,}6827$ (identique à la partie A, car c'est toujours « $P(-1\sigma<W<1\sigma)$ »).

2. $0{,}15 = 3\times0{,}05 = 3\sigma$: $P(|W|>0{,}15) \approx 0{,}0027$.

3. Les résultats sont exactement les mêmes qu'en partie B, malgré un σ différent : cela confirme que la règle empirique ne dépend que du nombre d'écarts-types considérés, pas de la valeur de σ elle-même (cohérent avec A.3).

Exercice 2

1. Écart-type de $x_{25} = 0{,}2\times\sqrt{25} = 0{,}2\times5 = 1$.

2. $x_{25} \rightsquigarrow \mathcal{N}(0{,}6\ ;\ 1)$. $P(0<x_{25}<1) = P((0-0{,}6)/1<Z<(1-0{,}6)/1) = P(-0{,}6<Z<0{,}4) \approx 0{,}381$ (calculatrice).

3. Les deux écarts-types (celui de x_{25} et celui de x_{100} en partie C) valent tous les deux 1 : les deux distributions sont donc aussi étalées l'une que l'autre. C'est cohérent : on a compensé un nombre d'étapes plus petit (25 au lieu de 100, soit 4 fois moins) par un bruit à chaque étape 4 fois plus fort en variance (0,04 au lieu de 0,01) : $25\times0{,}04 = 1 = 100\times0{,}01$, donc la variance finale est identique.

Exercice 3

1. Initialisation (n=0) : les deux poids valent 0, ils sont égaux. Hérédité : si à l'itération n les poids de A et de B sont égaux et reçoivent le même gradient, alors après la mise à jour (poids $\leftarrow$ poids $-\ \eta\times$gradient), ils restent égaux à l'itération n+1 (on soustrait la même quantité aux deux poids égaux). Par récurrence, les poids de A et de B restent égaux à toutes les itérations.

2. C'est problématique car les neurones A et B, censés apprendre à détecter des caractéristiques différentes dans les données, resteront à jamais des copies exactes l'un de l'autre : le réseau perd une grande partie de sa capacité de représentation, comme s'il n'avait, dans les faits, qu'un seul neurone au lieu de deux.

Notions mobilisées dans ce document

Hors-programme de terminale spécialité mathématiques

Loi normale, règle empirique (68–95–99,7 %), utilisation de la calculatrice pour une loi $\mathcal{N}(\mu\ ;\ \sigma^2)$.

Centrer-réduire une variable normale pour calculer une probabilité.

Ces notions ne figurent pas dans le programme actuel de terminale (BO spécial n°8 du 25 juillet 2019) : la partie « Probabilités » de ce programme s'arrête au schéma de Bernoulli, à la loi binomiale, aux sommes de variables aléatoires et à la loi des grands nombres.

Là où ces notions sont réellement enseignées

BTS, classes préparatoires, licences scientifiques : la loi normale y est incontournable, notamment pour l'inférence statistique et le traitement du signal.

Option mathématiques expertes : peut constituer un prolongement naturel pour les élèves visant ces filières.

Ancien programme de Terminale S (avant 2019) : la loi normale y figurait explicitement — ce document peut donc aussi servir de pont pour des élèves ou enseignants familiers de cet ancien programme.

Première (spécialité mathématiques) — notions réellement au programme, réutilisées ici

Raisonnement par récurrence (partie B et exercice 3).

Notion de variable aléatoire et de probabilité, sur lesquelles s'appuie la loi normale.