Projet de modélisation

Sujet : transports

Modélisation du fonctionnement d'une ligne de bus

Phase 1 : Modélisation du fonctionnement d'une ligne de bus

Mathis BOUKHELLOUF
Axelle HUGET
Esther MÉNARD

Partie 1 : Modélisation probabiliste des phases d'arrêts dans le cas d'un seul bus

(1.1) Personne ne descend, donc le nombre de passagers avant l'arrêt *Laurent Bonnevay – Astroballe* est la somme du nombre de passagers qui sont montés depuis l'arrêt *Vaulx-en-Velin – La Grappinière*, *i.e.* $L_{1,9} = \sum_{s=1}^{9} B_{1,s}$. Ainsi, par additivité de la loi de Poisson,

$$\boxed{L_{1,9} \rightsquigarrow \mathscr{P}(45)}$$

(1.2) Comme personne ne descend, on cherche $\mathrm{p}(L_{1,9} > 70)$. On peut appliquer le théorème central limite au vu de la valeur de λ, mais on propose ici de faire le calcul brut :
$\mathrm{p}(L_{1,9} > 70) = 1 - \mathrm{p}(L_{1,9} \leq 70) = 1 - \sum_{n=1}^{70} \mathrm{e}^{-45} \dfrac{45^n}{n!}$, soit :

$$\boxed{\mathrm{p}(L_{1,9} > 70) \simeq 0{,}02\%}$$

Selon ce modèle, il est très, très peu probable que des passagers n'aient pas pu monter dans le bus. On devrait en moyenne avoir 45 personnes à l'intérieur lorsqu'il arrive à *Laurent Bonnevay – Astroballe*, c'est l'espérance de la variable aléatoire $L_{1,9}$.

(1.3)
— $\{L_{1,9} = l\}$: « l passagers arrivent à *Laurent Bonnevay – Astroballe* ».
— Sachant que l passagers sont dans le bus, chacun de ces passagers a la probabilité μ_{10} de descendre. Il s'agit de la réalisation de l épreuves de Bernoulli indépendantes – *a priori* la descente d'un passager n'influence pas celle d'un autre passager – de paramètre μ_{10}.

$$\boxed{A_{1,10} \mid \{L_{1,9} = l\} \rightsquigarrow \mathscr{B}(l, \mu_{10})}$$

(1.4) Les $\{L_{1,9} = l\}_{l \in \mathbb{N}^*}$ forment un système complet d'évènements, la formule des probabilités totales donne alors, pour un nombre $m \leq l$ de passagers,

$$\begin{aligned}
\mathrm{p}(A_{1,10} = m) &= \sum_{l=1}^{+\infty} \mathrm{p}\left(A_{1,10} = m \,\middle|\, \{L_{1,9} = l\}\right) \mathrm{p}\left(\{L_{1,9} = l\}\right) \\
&= \sum_{l=1}^{+\infty} \binom{l}{m} {\mu_{10}}^m (1-\mu_{10})^{l-m} \frac{\lambda^l}{l!} \mathrm{e}^{-\lambda} \\
&\underset{m \leq l}{=} \sum_{l=m}^{+\infty} \frac{\cancel{l!}}{m!(l-m)!} \times \frac{1}{3^m} \times \frac{2^{l-m}}{3^{l-m}} \times \frac{45^l}{\cancel{l!}} \mathrm{e}^{-45} \\
&= \frac{\mathrm{e}^{-45}}{2^m m!} \sum_{l=m}^{+\infty} \frac{30^l}{m!(l-m)!} \underset{k=l-m}{=} \frac{\mathrm{e}^{-45}}{2^m m!} \sum_{k=0}^{+\infty} \frac{30^{k+m}}{k!} \\
&= \mathrm{e}^{-45} \frac{15^m}{m!} \sum_{k=0}^{+\infty} \frac{30^k}{k!} = \mathrm{e}^{-45} \frac{15^m}{m!} \times \mathrm{e}^{30} \\
&= \mathrm{e}^{-15} \frac{15^m}{m!}
\end{aligned}$$

Ainsi,

$$\boxed{A_{1,10} \rightsquigarrow \mathscr{P}(15)}$$

Partie 2 : Mise en équation et analyse théorique du système dans le cas de plusieurs bus

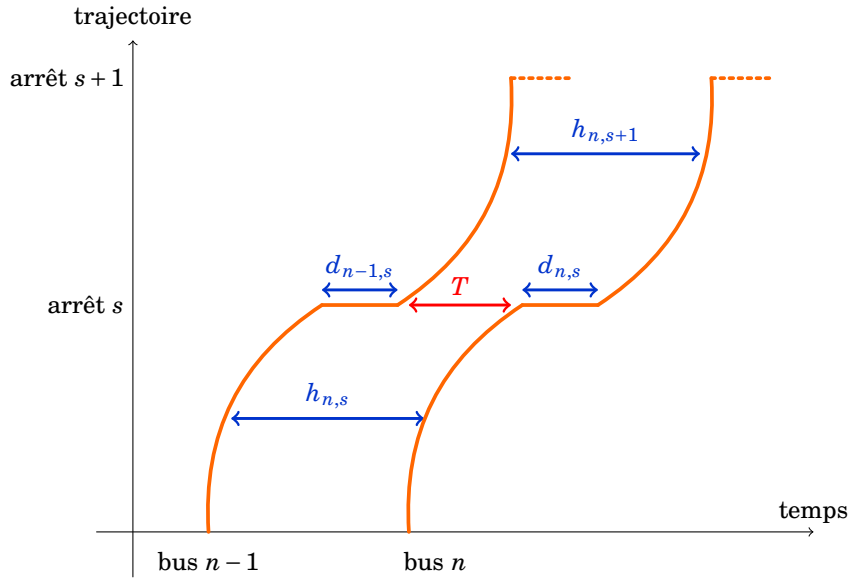
(2.1) Le bus ouvre ses portes, puis les $B_{n,s}$ personnes entrent chacun à leur tour, avant que le bus ne ferme ses portes. D'où :

$$\boxed{d_{n,s} = c + B_{n,s}b}$$

(2.2) Pendant le temps $h_{n,s}$, le nombre de personnes qui arrivent attendre le bus augmente suivant la fréquence λ_s, d'où :

$$\boxed{B_{n,s} = \lambda_s h_{n,s}}$$

(2.3) On suppose le temps de parcours T constant, c'est-à-dire que quels que soient le bus n et l'arrêt s, $T = h_{n,s+1} - d_{n,s} = h_{n,s} - d_{n-1,s}$, ce qu'on peut le constater sur le graphique ci-dessous :



(2.1) et (2.2) donnent $h_{n,s+1} \cancel{- c} - b\lambda_s h_{n,s} = h_{n,s} \cancel{- c} - b\lambda_s h_{n-1,s}$, soit $h_{n,s+1} = (1+b\lambda_s)h_{n,s} - b\lambda_s h_{n-1,s}$, ou encore :

$$\boxed{h_{n,s+1} = h_{n,s} + b\lambda_s \left(h_{n,s} - h_{n-1,s}\right)}$$

(2.4)
- Si, à un arrêt s, l'écart entre les bus $n-1$ et n et n et $n+1$ est le même, alors l'écart entre les bus n et $n+1$ à l'arrêt suivant $s+1$ est le même que ces écarts : on a donc, localement, une ligne de bus fluide ;
- sinon, cette équation montre que la durée de montée des passagers et que la fréquence d'arrivée des passagers à un arrêt font varier les temps d'attente entre deux bus. Ce modèle aboutit donc à l'instabilité de la ligne de bus.

(2.5) En ajoutant les $\tau_{n,s}$ et $\tau_{n-1,s}$, il vient :

$$\boxed{h_{n,s+1} = h_{n,s} + (b\lambda_s - \alpha)\big(h_{n,s} - h_{n-1,s}\big)}$$

Il suffit d'avoir une valeur de α proche de $b\lambda_s$ pour rétablir localement la stabilité dans la ligne de bus.

(2.6) Notons $\bar{T}_{N,s}$ le temps d'attente moyen pour N bus à l'arrêt s :

— le temps d'attente moyen pour un bus, disons le numéro n, à l'arrêt s est donné par :

$$\boxed{\bar{T}_{1,s} = \frac{1}{h_{n,s}} \int_0^{h_{n,s}} t \,\mathrm{d}t = \frac{h_{n,s}}{2}}$$

— pour n bus successifs, on effectue la moyenne pondérée des temps d'attente pour chaque bus : on obtient $\bar{T}_{n,s} = \dfrac{\sum_{i=1}^{n} \cancel{\lambda_s} h_{i,s} \times \dfrac{h_{i,s}}{2}}{\sum_{i=1}^{n} \cancel{\lambda_s} h_{i,s}} \simeq \dfrac{1}{2nH} \sum_{i=1}^{n} {h_{i,s}}^2$, l'approximation se justifiant par le fait qu'en moyenne le passage entre deux bus successifs vaut l'écart théorique H.

Or, $\mathrm{V}(h_{n,s}) := \dfrac{1}{n}\sum_{i=1}^{n}\big(h_{i,s} - H\big)^2 = \dfrac{1}{n}\sum_{i=1}^{n} {h_{i,s}}^2 - \dfrac{2H}{n}\sum_{i=1}^{n} h_{i,s} + H^2 = \dfrac{1}{n}\sum_{i=1}^{n} {h_{i,s}}^2 - H^2$, ce qui donne $\sum_{i=1}^{n} {h_{i,s}}^2 = nH^2 + n\mathrm{V}(h_{n,s})$, d'où :

$$\boxed{\bar{T}_{n,s} = \frac{H}{2} + \frac{\mathrm{V}(h_{n,s})}{2H}}$$

(2.7) — Si tous les bus respectent l'écart théorique H, il est facile de vérifier que $\bar{T}_{n,s} = \dfrac{H}{2}$;

— si les bus voyagent deux par deux, l'écart entre chaque bus n et $n+1$, puis $n+2$ et $n+3$ est nul, tandis que l'écart entre $n+1$ et $n+2$ ne change pas : pour simplifier, tout se passe comme si on enlevait les bus pairs. Dans ce cas, l'écart temporel entre deux paires de bus vaut $2H$. Il vient alors $\bar{T}_{n,s} = H$.

La construction $I_s = \dfrac{\bar{T}_{n,s} - H/2}{H/2}$ convient alors aux deux cas précédents. En développant le calcul, on obtient :

$$\boxed{I_s = \frac{\mathrm{V}(h_{n,s})}{H^2}}$$

Partie 3 : Implémentation informatique du modèle et test du système

(3.1) Les fonctions sont en annexe. Quelques remarques cependant :

— les **bibliothèques et fonctions** suivantes ont été importées pour **toute la partie 3** :

```
from numpy import zeros, var, mean
from numpy.random import poisson, uniform
from matplotlib.pyplot import xlabel, ylabel, title, grid, plot, hist
```

On rappelle aussi que pour Python, les listes et matrices sont indexés à partir de 0 et pas 1 ;

— en plus des fonctions demandées, on a jugé nécessaire d'implémenter une fonction renvoyant une matrice $N \times S$ ayant pour coefficient (n,s) la durée $h_{n,s}$. En effet, implémenter directement le temps séparant l'arrivée de deux bus successifs nécessiterait de coder une fonction récursive, qui consommerait énormément de mémoire pour le calcul des derniers coefficients. Aussi, pour le premier bus, on a convenu de généraliser à tous les arrêts $h_{1,s} = H$ dans le cas d'une demande en passagers constante, car on ne connaît pas la demande des passagers avant le démarrage de la ligne ;

— des **tests basiques** ont été réalisés :

— lorsqu'il n'y a aucun passager, tous les bus mettent S fois le temps de parcours pour parcourir la ligne ;

— on s'est rendu compte, pour des flux de passagers élevés, que lorsque le bus $n+1$ dépasse le bus n... Le bus $n+1$ se met à remonter le temps ! Cela se comprend par le fait que les deux bus permutent leurs numéros, le signe de $h_{n,s+1}$ devient négatif et les formules de la partie 2 ne sont alors plus valables. Si on considère qu'il n'y a pas de dépassement des bus, on arrive en fait au cas extrême de la paire de bus. Au vu des paramètres donnés pour les applications numériques, ce cas n'arrive pas avec le présent modèle ;

— quelques remarques quant au **déplacement des bus**, à voir peut-être pour la phase 2 :

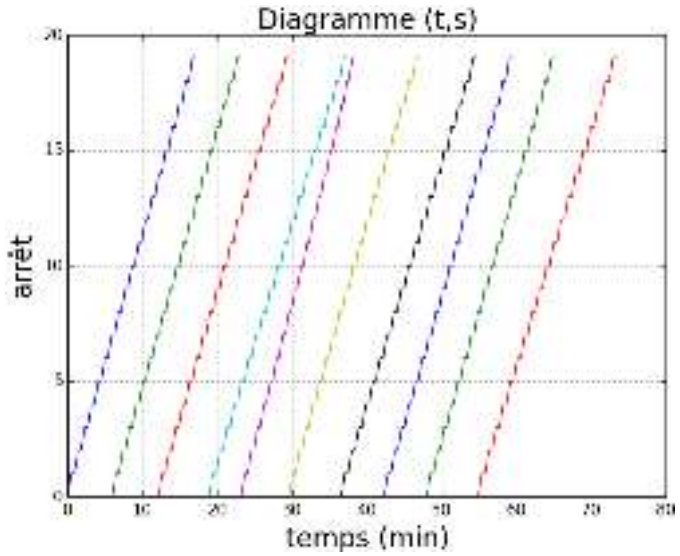
— on considère ici qu'il n'y a pas dépassement, c'est la logique avancée dans les formules et les indicateurs utilisés, par exemple $I_s = 1$ lorsque les bus voyagent par paire, c'est le cas extrême. Cependant, on peut tout à fait imaginer le cas où les conducteurs ne veulent pas rester à la traîne et avancer. Dans ce cas, il suffit de permuter les bus concernés au moment du dépassement - ce qui est en revanche un peu moins évident au niveau de la programmation ;

— on peut aussi affiner le modèle en implémentant des feux rouges, et ainsi modifier le temps de parcours ; ou bien en modifiant λ_s à tous les arrêts : pour le C3 par exemple, beaucoup plus de personnes montent à *Vaulx – Hôtel de Ville - Campus* qu'à *Lefèvre*.

(3.2) S'il existe un bus n tel que pour tout arrêt s $h_{n,s} = H$, alors les temps d'attente suivants entre deux bus valent H. Il suffit alors que le premier bus ne prenne aucun retard pour que l'ensemble du système soit régulé... À condition que tous les paramètres soient déterministes.

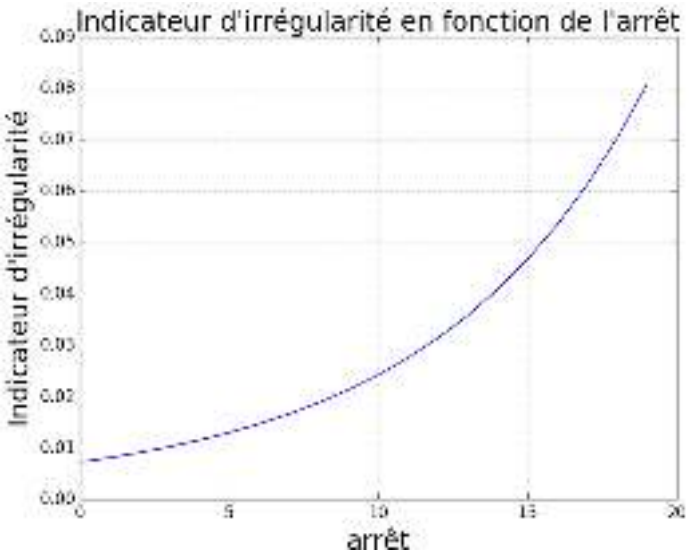
(3.3) On exécute le code ci-dessous dans l'éditeur, ce qui génère le diagramme ci-après :

```
>>> Diagrammets(10,20,3/60,6/60,0.5,6,1,True,False,[],0)
```



On exécute ensuite le code ci-dessous pour générer la courbe de l'indicateur d'irrégularité [1] :

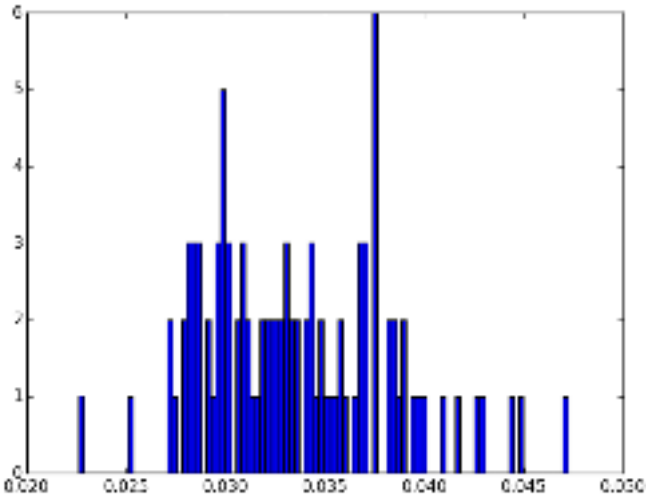
```
>>> CourbeII(10,20,6,3/60,1,True,[],0)
```



L'irrégularité augmente au fil des arrêts, passant d'environ 1% à l'arrêt de départ à 8% au terminus.

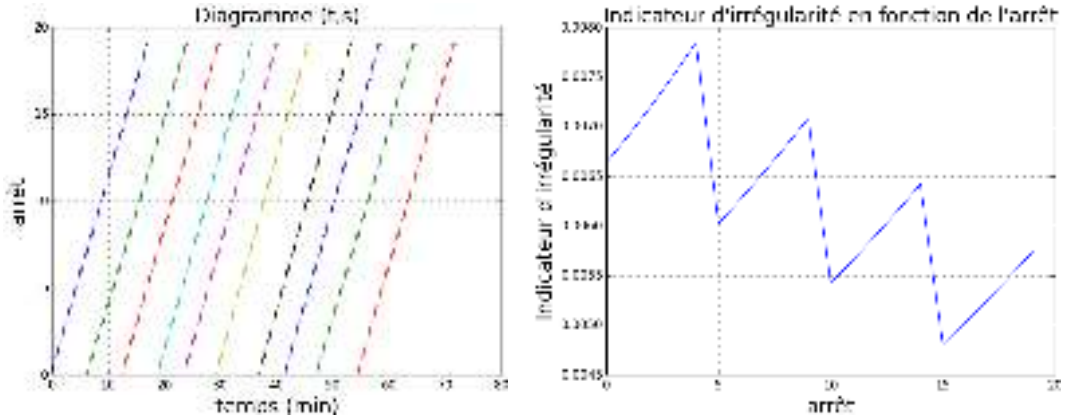
1. En fait, pas vraiment : comme la stochasticité change la date de départ à chaque génération, pour relier le diagramme (t,s) et la courbe de l'indicateur d'irrégularité, il faudrait en toute rigueur utiliser les mêmes $h_{n,s}$ et les mêmes dates de départ. On a donc généré les mêmes dates de départ et la `MatriceH` en découlant pour les deux courbes de cette page.

(3.4)
```
>>> Histogramme(10,20,6,3/60,1,True,[4,9,14],0)
```



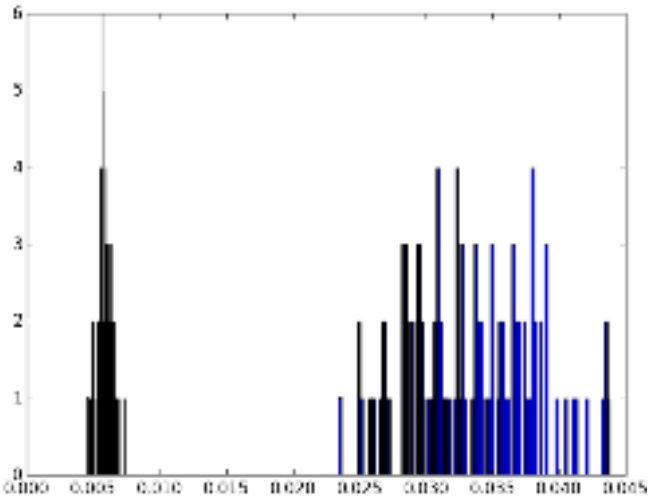
L'indicateur d'irrégularité varie d'une instance à une autre, en étant centré dans un intervalle allant d'environ 2 à 5%.

(3.5)
```
>>> Diagrammets(10,20,3/60,6/60,0.5,6,1,True,False,0.4,[4,9,14])
>>> CourbeII(10,20,6,3/60,1,True,[4,9,14],0.4)
```



La régulation s'observe assez bien juste après les arrêts considérés, comme le montre la courbe de l'indicateur d'irrégularité, ce qui coïncide avec le commentaire fait en (2.5).

(3.6)
```
>>> Histogramme(10,20,6,3/60,1,True,[],0)
>>> Histogramme(10,20,6,3/60,1,True,[4,9,14],0.4)
```



Lorsque la régulation est effective, on remarque que les indicateurs d'irrégularité sont centrés autour de 0,6%, c'est moins que les 2 à 5% environ obtenus sans régulation. De plus, ces indicateurs sont plus serrés, comme si on avait une loi normale de plus petit écart-type : on en déduit que cette stratégie est plus fidèle.

Annexe : codes de la partie 3

Pour éviter de rendre cette annexe rébarbative, les codes ci-joints sont ceux issus de la stratégie de régulation, mais il suffit de prendre `alpha` = 0 et `LAR`[2] = `[]` pour avoir la version non régulée demandée aux questions (3.1) et (3.3).

```
def TempsArret(n, s, c, b, ls, M, stoT):
    ''' Retourne le temps d'arrêt à l'arrêt s pour le bus n. ls est le flux
        de passagers ; M est une matrice qui est, dans le code de la
        simulation, la matrice des h(n,s) ;  stoT est un booléen retournant
        True si l'arrivée des passagers est stochastique. '''
    if stoT == True: # si c'est stochastique, le nombre de passagers par
        minute suit une loi de Poisson
        ls = poisson(ls)
    return c + ls*M[n][s]*b
```

```
def DateDepart(n, H, stoD):
    ''' Détermine la date de départ du bus n au début de la ligne. Chaque bus
         part théoriquement un temps H après le dernier ; stoD est un booléen
         déterminant si cette date est stochastique. '''
    if n == 0:
        return 0 # la base des temps est le départ du premier bus
    else:
        if stoD == True:
            return n*H + uniform(-1,1)
        else:
            return n*H
```

```
def MatriceH(N, S, H, b, ls, stoD, LAR, alpha):
    ''' Regroupe les résultats de la suite h_(n,s) dans une matrice N lignes,
         S colonnes.'''
    MatH = zeros((N,S))
    for s in range(S):
        MatH[0][s] = H # conditions initiales sur le premier bus
    for n in range(N):
        MatH[n][0] = DateDepart(n,H,stoD) - (n-1)*H # initialisation ; la
            soustraction vient du fait que h(n,s) est un écart relatif
        for s in range(0,S-1):
            if s in ListeArretsRegules: # s'il y a régulation
                MatH[n][s+1] = MatH[n][s] + (b-alpha)*ls*(MatH[n][s]-MatH[n-1
                    ][s])
            else:
                MatH[n][s+1] = MatH[n][s] + b*ls*(MatH[n][s]-MatH[n-1][s])
    return MatH
```

2. pour « liste d'arrêts régulés ».

```
def IndicateurIrregularite(N, S, s, H, b, ls, stoD, LAR, alpha):
    ''' Retourne l'indicateur d'irrégularité à un arrêt s d'une ligne en
        comportant S. La variance est calculée pour les N bus de la ligne.
        '''
    MatH = MatriceH(N,S,H,b,ls,stoD,LAR,alpha)
    return (1/H**2)*var([MatH[n][s] for n in range(N)])
```

```
def SimulationTrajetsBus(N, S, b, c, parcours, H, ls, stoD, stoT, LAR,alpha):
    ''' Simule le parcours de N bus sur une ligne de S arrêts. Le temps de
        parcours est constant, les départs et temps d'arrêt peuvent, ou non,
        être stochastiques, information détenue par les booléens stoD & stoT.
        '''
    Depart = zeros((N,S))
    Arrivee = zeros((N,S))
    MatH = MatriceH(N,S,H,b,ls,stoD,LAR,alpha)
        for i in range(0,N):
        Depart[i][0] = DateDepart(i,H,stoD)
        Arrivee[i][0] = Depart[i][0]
            for j in range(1,S):
            Depart[i][j] = Depart[i][j-1] + parcours + TempsArret(i,j,c,b,ls,
                MatH,stoT)
            Arrivee[i][j] = Depart[i][j-1] + parcours
    return Depart, Arrivee
```

```
def Diagrammets(N, S, b, c, parcours, H, ls, stoD, stoT, LAR, alpha):
    ''' Affiche le diagramme (t,s) d'une simulation de trajets de bus. '''
    Arrets = [s for s in range(0,S)]
    ArretsTries = sorted([s for s in range(0,S)]*2) # liste de doublons : il
        y a une arrivée puis un départ à chaque arrêt, ce qui permettra de
        remplir une matrice de temps pour tracer le diagramme
    xlabel("temps (min)", fontsize=20)
    ylabel("arrêt", fontsize=20)
    title("Diagramme (t,s)", fontsize=20)
    Depart, Arrivee = SimulationTrajetsBus(N, S, b, c, parcours, H, ls, stoD,
         stoT, LAR, alpha)
    Temps = [] # contient les temps de départ et d'arrêt
    for n in range(0,N):
        for s in Arrets:
            Temps.append(Arrivee[n][s])
            Temps.append(Depart[n][s]) # chronologiquement, le bus arrive à l
                'arrêt avant d'en partir
        plot(Temps,ArretsTries)
        Temps = [] # on vide la liste pour calculer les temps du prochain bus
    grid(True)
```

```
def CourbeII(N ,S, H, b, ls, stoD, LAR, alpha):
    ''' Affiche la courbe de l'indicateur d'irrégularité à chaque arrêt. '''
    MatH = MatriceH(N,S,H,b,ls, stoD, LAR, alpha)
    II = [(1/H**2)*var([MatH[n][s] for n in range(N)]) for s in range(S)]
    xlabel("arrêt", fontsize=20)
    ylabel("Indicateur d'irrégularité", fontsize=20)
    title("Indicateur d'irrégularité en fonction de l'arrêt", fontsize=20)
    plot(range(S),II)
    grid(True)
```

```
def Histogramme(N, S, H, b, ls, stoD, LAR, alpha):
    ''' Affiche l'histogramme de la moyenne de l'indicateur d'irrégularité,
        pour 100 générations du modèle. '''
    Resultats = []
    for i in range(100):
        Resultats.append(mean([IndicateurIrregularite(N,S,s,H,b,ls,stoD,LAR,
            alpha) for s in range(S)]))
    hist(Resultats,bins=100,normed=0)
```