Traitement du sujet par CoffeeScript

Comme à l'accoutumée dans cette rubrique, l'outil ***alcoffeethmique***[1] sera utilisé. Les spécificités de ce langage qui seront utilisées seront les suivantes :

1. la possibilité de placer le test après l'action réalisée (« rajouter 4 ml s'il reste moins de 5 ml » au lieu de « s'il reste moins de 5 ml, rajouter 4 ml ») ;
2. la possibilité de manipuler les suites comme listes de nombres, en poussant les termes successifs de la suite à la fin de la liste ; ce qui permet ensuite de représenter graphiquement la suite ou de faire des statistiques dessus.

I/ Premier protocole

Dans un premier temps, on n'injecte le médicament qu'une fois au début. La suite décrite ainsi est donc géométrique puisque

> On effectue à l'instant 0 une injection de 10 mL de médicament. On estime que 20% du médicament est éliminé par minute. Pour tout entier naturel n, on note u_n la quantité de médicament, en mL, restant dans le sang au bout de n minutes. Ainsi $u_0=10$.

Le premier terme de la suite étant 10, on initialise la suite u à un tableau ne comprenant que 10, ce qui se fait avec u=[10]. Ensuite, dans la boucle sur l'indice n, on multiplie la variable med[2] par 0,8 et on pousse le résultat à la suite de u :

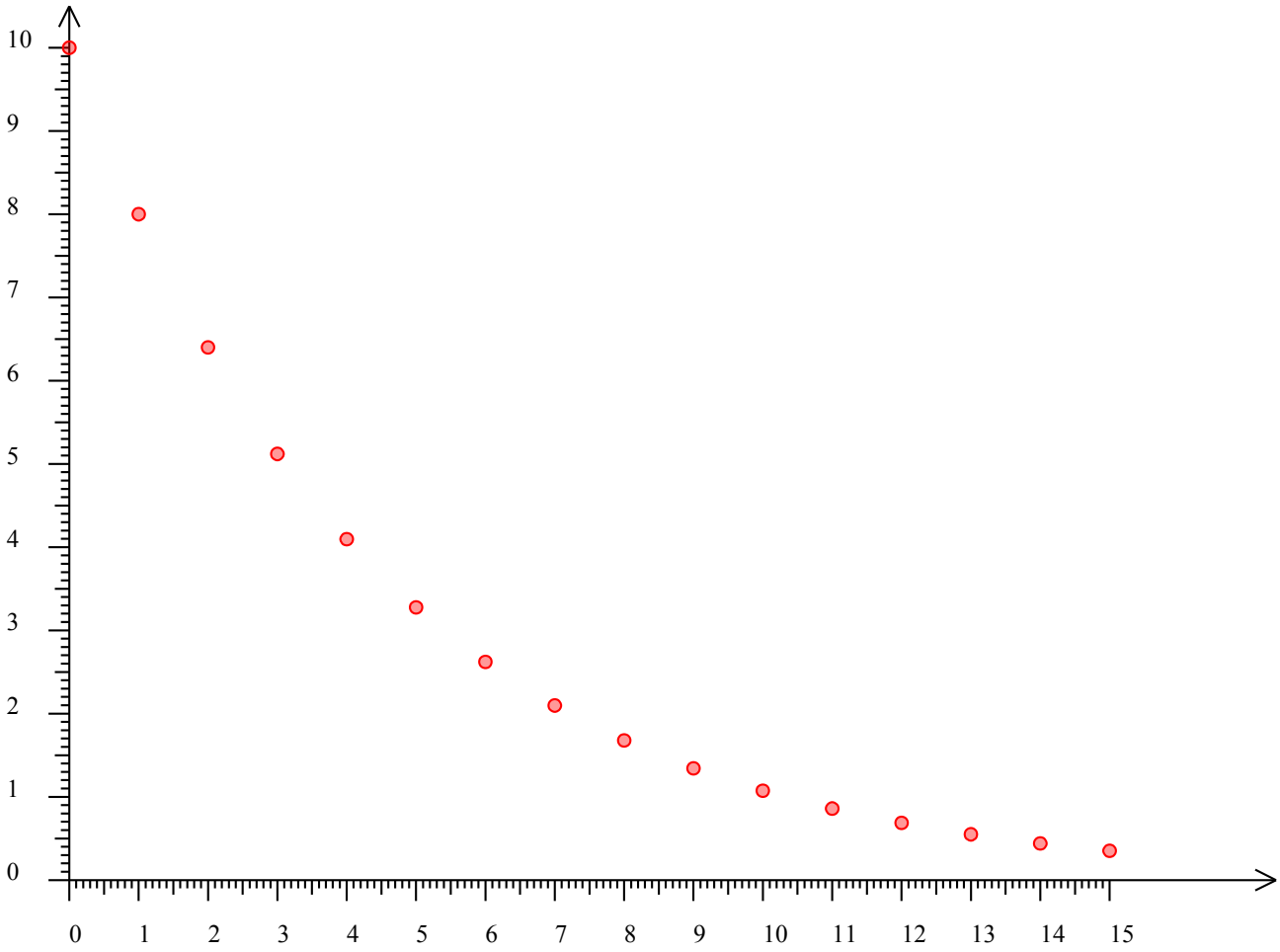
```
u = [10]
for n in [1..15]
    med = u[n-1]*.8
    u.push med

dessineSuite u, 15, 0, 10, 3, 'red'
```

1 Normalement joint à ce document

2 Cette variable contient la quantité de médicament actuellement présente dans le corps du patient et s'appelle donc med.

L'exécution du script produit la représentation graphique de la suite :



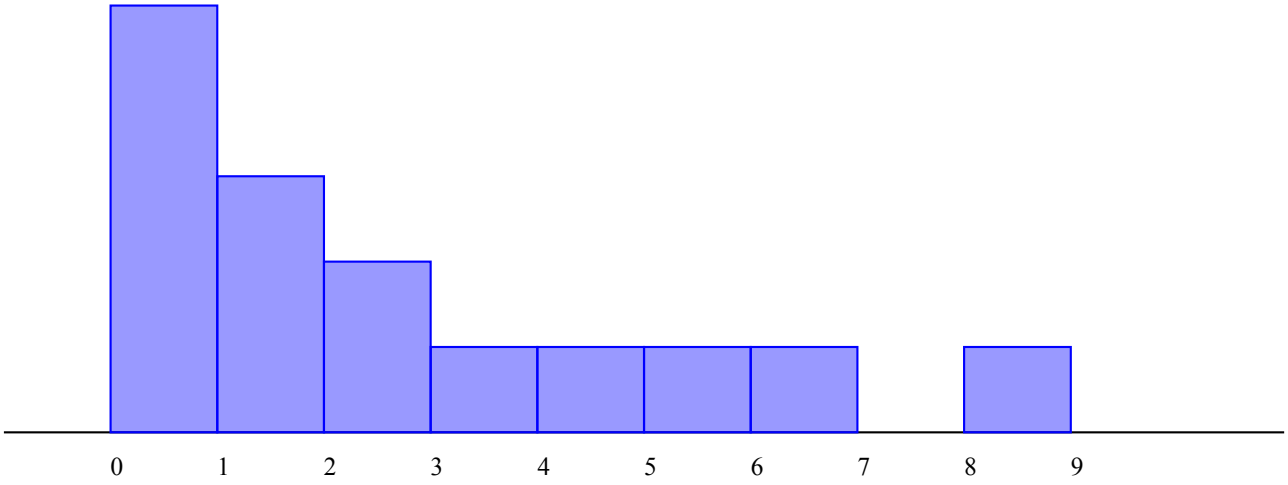
On voit que, la suite tendant vers 0, la quantité de médicament est loin d'être constante. Il faut donc, pour améliorer l'efficacité de celui-ci, faire de temps en temps d'autres injections. Ce sera l'objet des autres protocoles étudiés dans le problème.

Pour illustrer le fait que pendant la plupart du temps, le patient est peu soigné, on peut, au lieu de représenter graphiquement la suite, dessiner plutôt un histogramme, par la variante du script que voici :

```
u = [10]
for n in [1..15]
    med = u[n-1]*.8
    u.push med

histogramme u, 0, 10, 10, 10
```

L'histogramme va de 0 à 10 en 10 valeurs et la hauteur des rectangles est 10. Certes ça fait beaucoup de 10… L'histogramme montre bien la disparité des concentrations au cours du temps :



II/ Second protocole

Cette fois-ci,

> Une machine effectue à l'instant 0 une injection de 10 mL de médicament.
> On estime que 20% du médicament est éliminé par minute. Lorsque la quantité de médicament tombe en-dessous de 5 mL, la machine réinjecte 4 mL de produit.

Ainsi, la seule différence par rapport au protocole précédent est qu'il faut rajouter une injection conditionnelle, la dernière phrase de l'énoncé se traduisant en CoffeeScript par

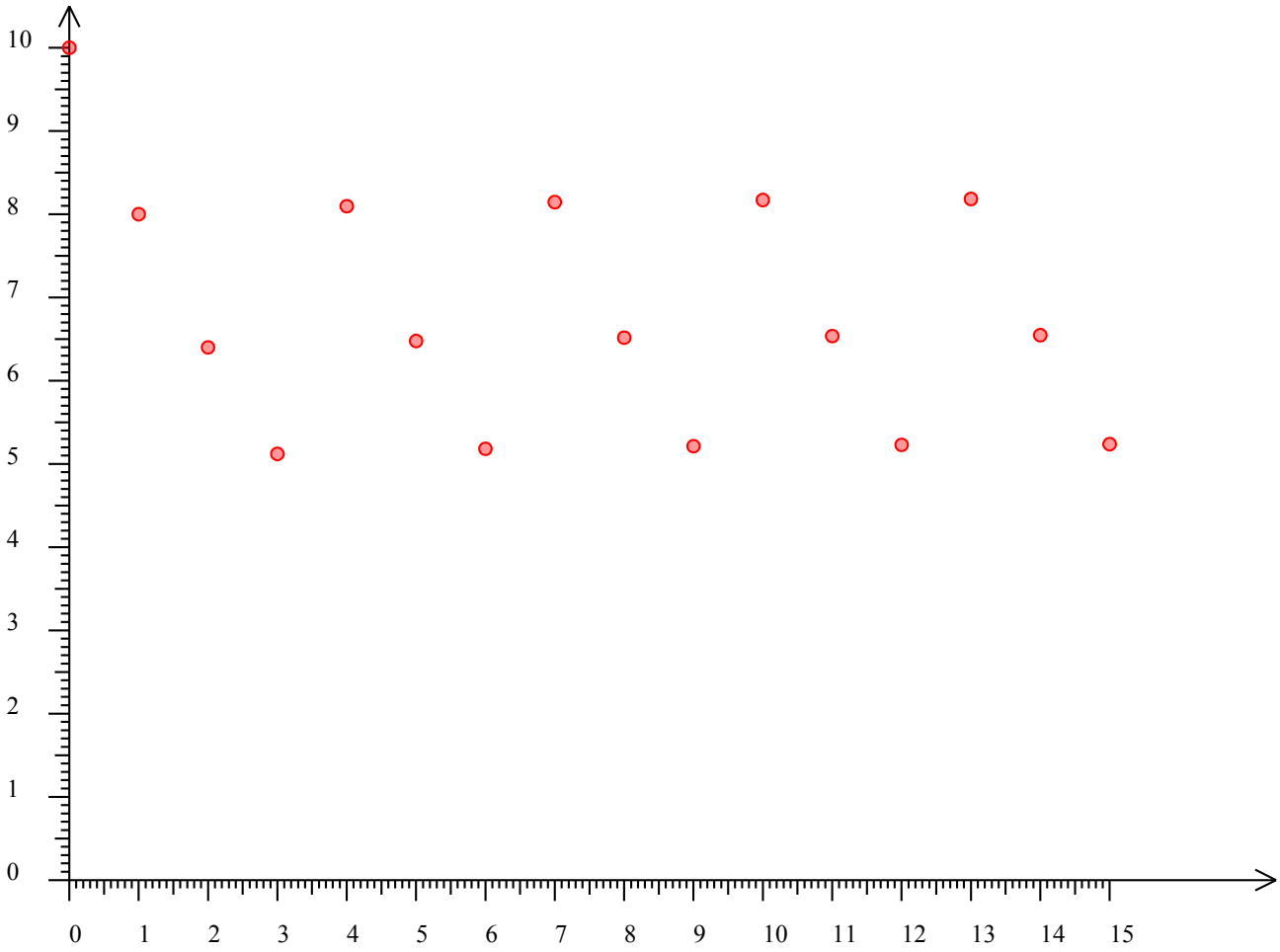
```
med += 4 if med < 5
```

Le script s'écrit donc

```
u = [10]
for n in [1..15]
    med = u[n-1]*.8
    med += 4 if med < 5
    u.push med

dessineSuite u, 15, 0, 10, 3, 'red'
```

La représentation graphique de la suite montre plus de constance (mais un peu de chaos) dans la concentration de médicament :



La suite est presque 3-périodique, à ceci près que les suites u_{2p}, u_{2p+1} et u_{2p+2} grandissent lentement.

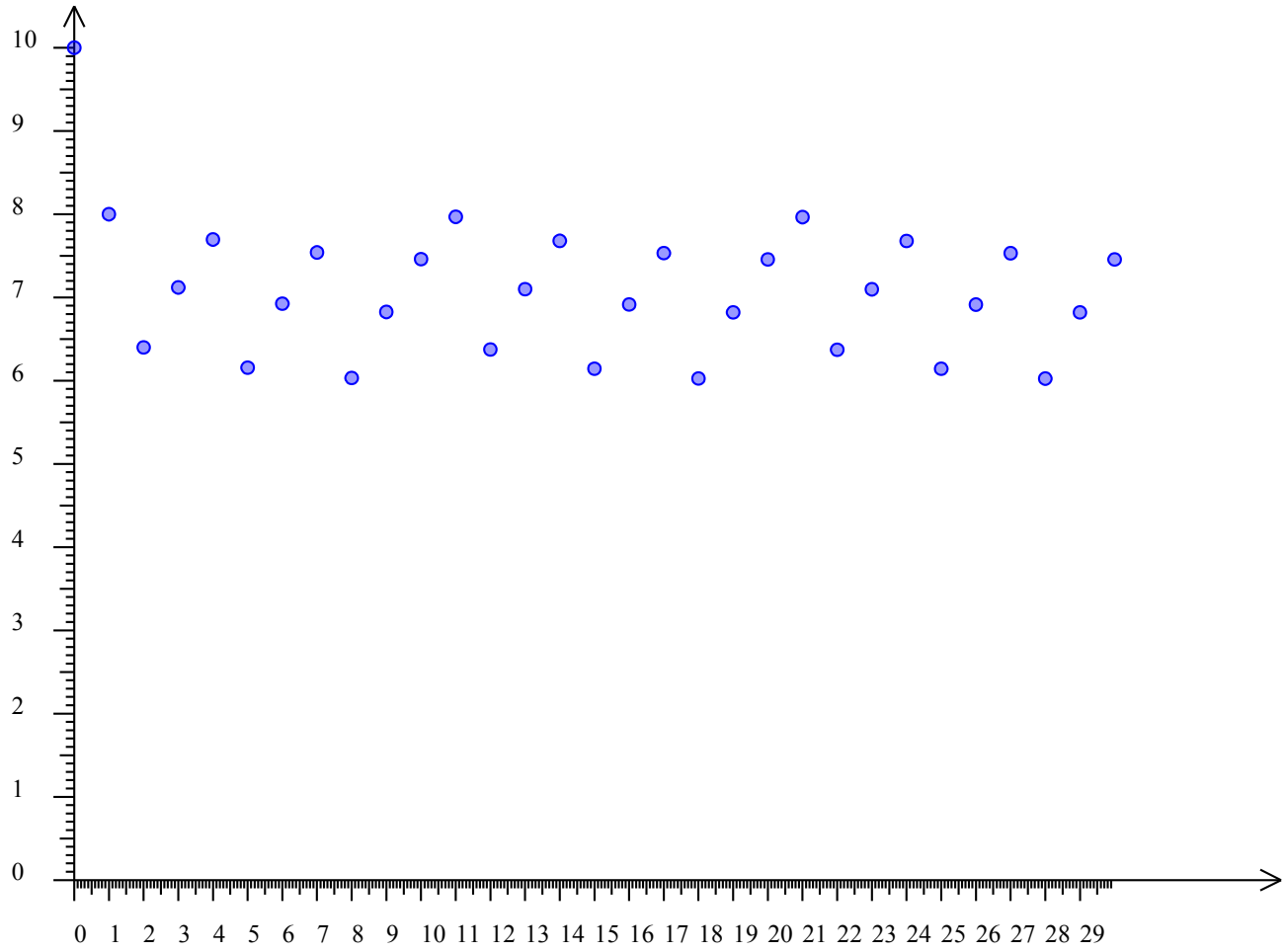
Une variante plus chaotique est obtenue par la suggestion de l'énoncé :

> On souhaite programmer la machine afin qu'elle injecte 2 mL de produit lorsque la quantité de médicament dans le sang est inférieure ou égale à 6 mL et qu'elle s'arrête au bout de 30 minutes.

Il suffit de modifier le script précédent en remplaçant 4 par 2 et 5 par 6 (ainsi que la taille de l'échantillon) :

```
u = [10]
for n in [1..30]
    med = u[n-1]*.8
    med += 2 if med < 6
    u.push med
dessineSuite u, 30, 0, 10, 3, 'blue'
```

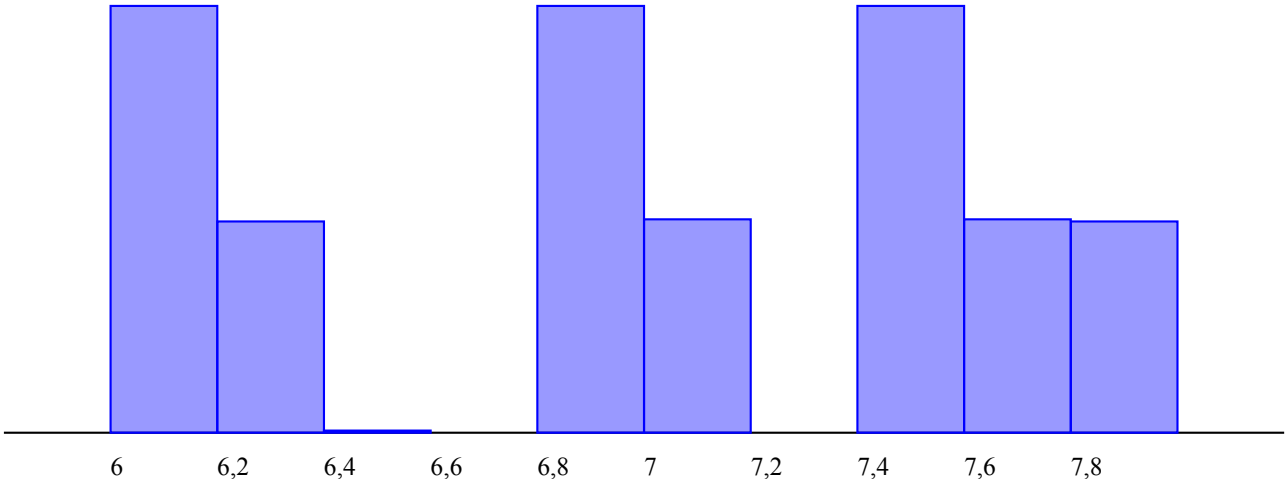
Pour changer, on a dessiné les points en bleu. La représentation graphique est nettement plus chaotique :



Ceci donne envie de représenter l'histogramme de cette suite, sur 1000 valeurs (entre 6 et 8, toujours avec 10 rectangles) :

```
u = [10]
for n in [1..1000]
    med = u[n-1]*.8
    med += 2 if med < 6
    u.push med
histogramme u, 6, 8, 10, 400
```

Voici l'histogramme :



Retour au second protocole, avec le calcul de la quantité totale de médicament injectée. Là encore, il y a peu à modifier dans le script précédent :

```
u = [10]
for n in [1..15]
    med = u[n-1]*.8
    med += 4 if med < 5
    u.push med

affiche laSommeDe u
```

L'exécution de cette variante produit l'effet suivant :

```
Algorithme lancé
109.05100102172673

Algorithme exécuté en 65 millisecondes
```

Ceci répond à la question…

III/ Troisième protocole

Pour avoir une quantité de médicament plus constante, on injecte constamment la même (faible) quantité de médicament :

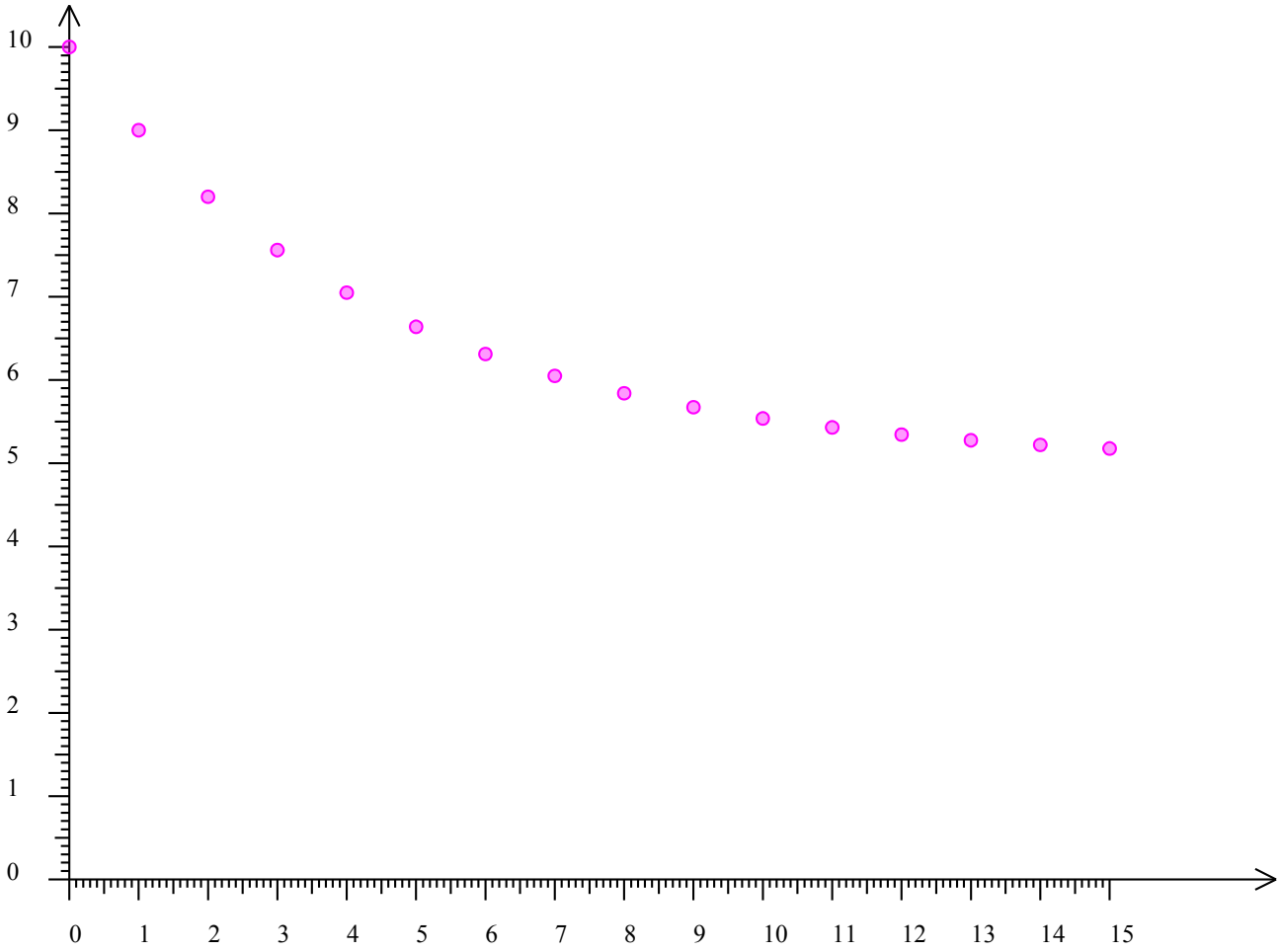
On programme la machine de façon que :
— à l'instant 0, elle injecte 10 mL de médicament,
— toutes les minutes, elle injecte 1 mL de médicament.
On estime que 20% du médicament présent dans le sang est éliminé par minute.

Avec ce nouveau protocole, la suite est arithmético-géométrique de raison inférieure à 1 et converge donc, vers une valeur qu'on va déterminer plus bas.

La représentation graphique de la suite se fait avec ce script :

```
u = [10]
for n in [1..15]
    med = u[n-1]*.8
    med++
    u.push med
dessineSuite u, 15, 0, 10, 3, 'magenta'
```

La représentation graphique montre la convergence :



La limite x est solution de l'équation 0,8x+1=x et vaut donc 5.

Annexe : Systèmes dynamiques

En fait, on étudie une suite récurrente, de la forme $u_{n+1}=f(u_n)$ où f est une fonction

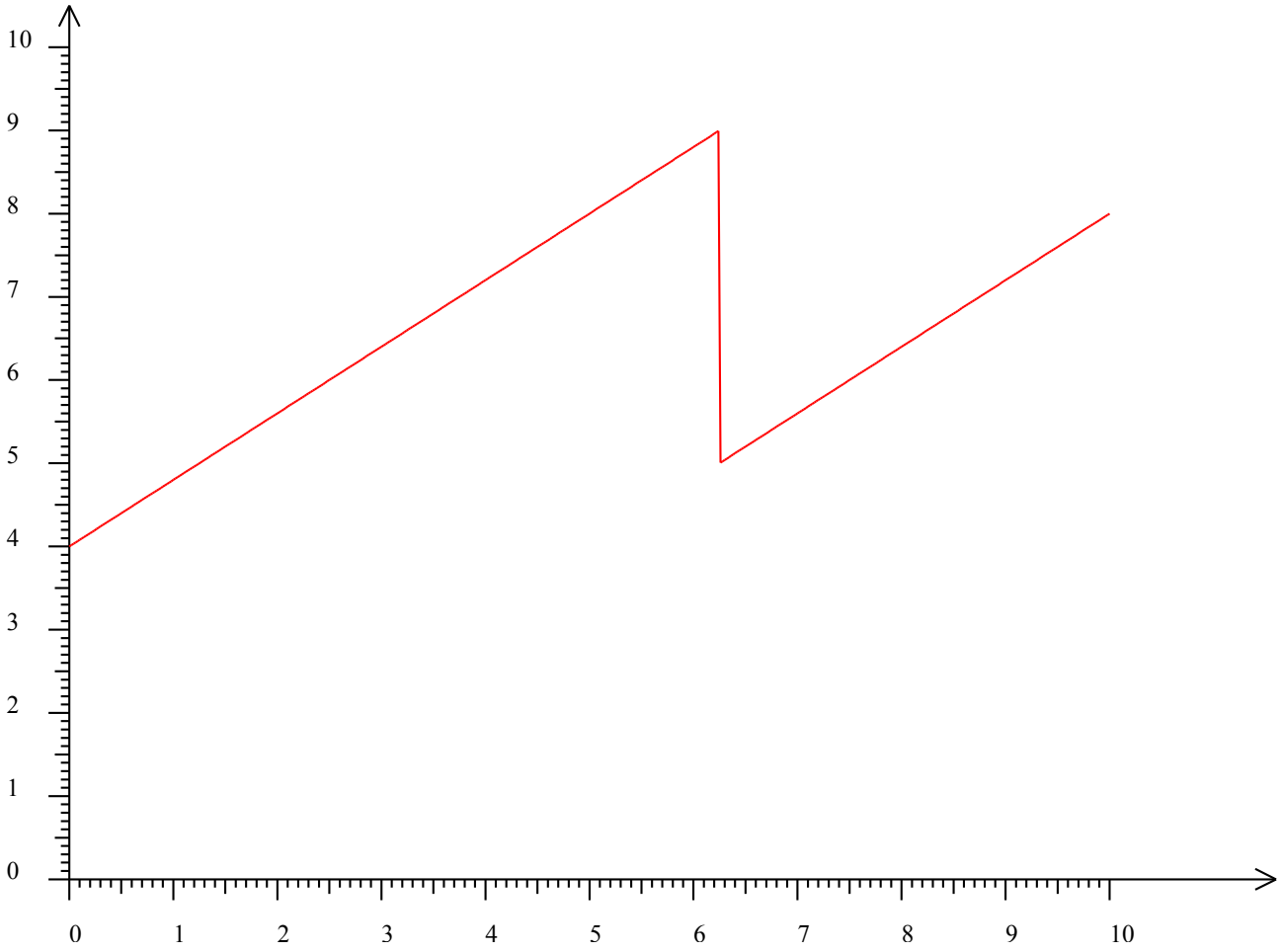
- linéaire dans le premier protocole (et dans ce cas la suite tend vers 0)
- affine par morceaux dans le second protocole (et, par discontinuité, la suite ne converge pas)
- affine dans le dernier protocole (on a vu qu'alors la suite converge).

Pour savoir si une suite de la forme $u_{n+1}=f(u_n)$ est convergente, on regarde la dérivée de f en chaque solution de l'équation f(x)=x : Si cette dérivée est inférieure à 1, la suite converge, sinon, elle ne converge pas. Or, dans le second protocole, la fonction n'est même pas continue là où sa représentation graphique « croise » la droite d'équation y=x. Pour représenter graphiquement la fonction f, on peut écrire ce script :

```
f = (x) ->
    if 0.8*x <  5
        0.8*x + 4
    else
        0.8*x

dessineFonction f, 0, 10, 0, 10
```

La représentation graphique est clairement discontinue :



Pas étonnant alors que la suite soit chaotique !

Alain Busser
Lycée Roland-Garros
Irem de La Réunion
Le Tampon