TP2

May 21, 2020

1 ST4 MDS - Données et Statistiques en Finance TP 2 : Optimal learning

1.1 Martin Bridoux & Karim El Asmar

[34]:
```
import numpy as np
import matplotlib.pyplot as plt
from math import *
from statsmodels.distributions.empirical_distribution import ECDF
import powerlaw
from IPython.display import clear_output
```

1.2 Partie 1

Définition des variables

[35]:
```
T = 10000
```

Définition du couple (σ, α) et simulation r_t

[36]:
```
s = 1  #sigma
a_0 = 2
r_0 = 1
e = np.random.normal(scale=sqrt(s), size=T)


def simulation(a_0):
    rendement = [r_0]
    alpha = [a_0]
    for t in range(1, T):
        rendement.append((a_0 - alpha[t - 1]) * rendement[t - 1] + e[t])
        alpha.append(rendement[t] / rendement[t - 1] + alpha[t - 1])
    return (rendement, alpha)
```

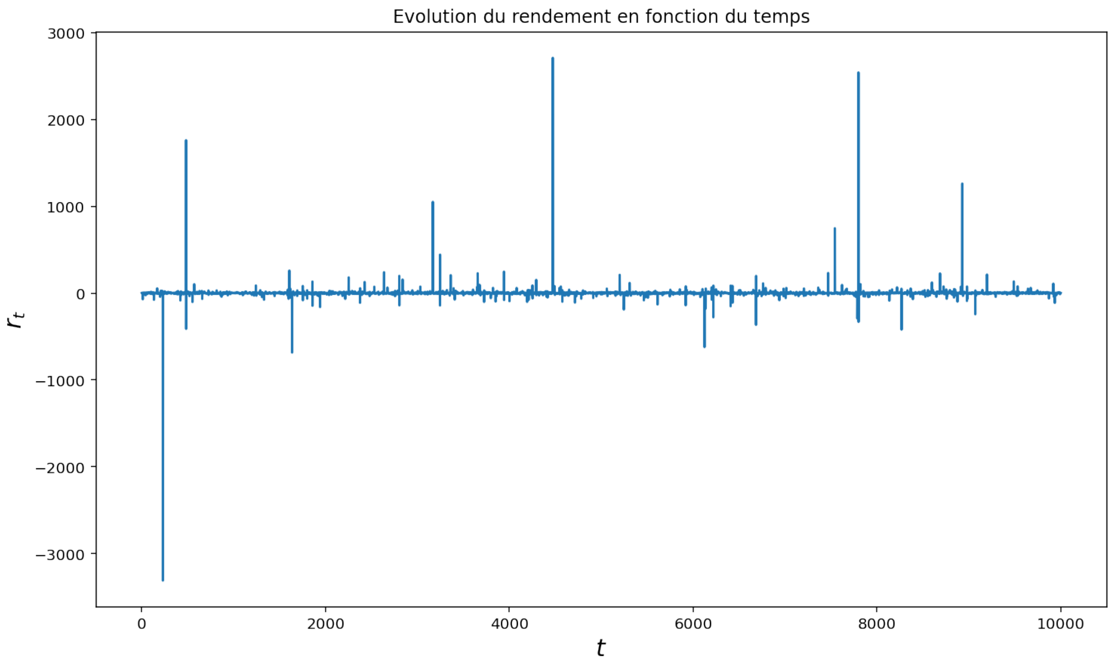
- Tracer

$$r_t$$

en fonction du temps

[37]:
```
figure = plt.figure()#figsize=(10,3))
ax1 = figure.add_subplot(111)
plt.plot(simulation(a_0)[0])
plt.xlabel("$t$", fontsize=16)
plt.ylabel("$r_t$", fontsize=16)
plt.title('Evolution du rendement en fonction du temps')
plt.show()
```

[37]:



Interprétation :

Les rendements présentent quelques pics, ce qui est cohérent avec les marchés financiers qui sont fondamentalement instables: par exemple, en temps de crise, les rendements chutent.
En pratique, la méthode utilisée est instable pour α>1.

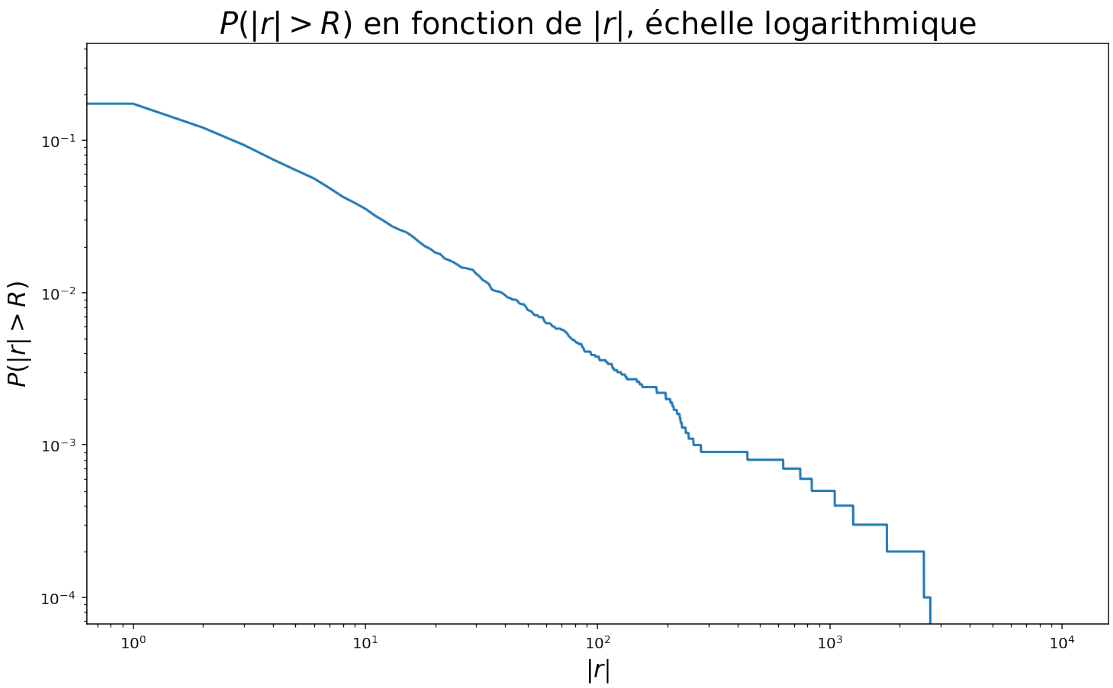
1.3 Partie 2

- Représenter $P(|r| > R)$.

[38]:
```
ecdf = ECDF(simulation(a_0)[0])
Y = 1 - ecdf(np.arange(1, T, 1))
```

```
fig = plt.figure()#figsize=(16,5))
plt.plot(Y)
plt.xscale('log')
plt.yscale('log')
plt.title('$P(|r|>R)$ en fonction de $|r|$, échelle logarithmique',fontsize=20)
plt.ylabel('$P(|r|>R)$',fontsize=16)
plt.xlabel('$|r|$', fontsize=16)
plt.show()
```

[38]:



Interprétation :

À partir de $|r| = 10^1$, l'allure de la courbe log-log est une droite. Cela signifie que $\mathbb{P}(|r| > R)$ et $|r|$ sont liés par une relation de puissance. Pour $|r| > 10^2$, la probabilité d'occurrence des évènements est très faibles.
On peut donc restreindre notre étude à certaines valeurs de r (par exemple : $|r| < 10^2$).

- Calculer l'exposant de queue de $P(|r|) \propto |r|^{-\gamma}$

[39]:
```
mypl = powerlaw.Fit(np.abs(simulation(a_0)[0]))
clear_output(wait=True)
mypl.alpha
```

[39]: 1.9984552734058718

Interprétation :

La distribution de P(r) est comparable à celle d'une *heavy-tailed*. Pour ce type de distribution, $\gamma \approx 2$, le résultat obtenu est donc cohérent.
De plus, cette valeur se rapproche de 2 quand T augmente.

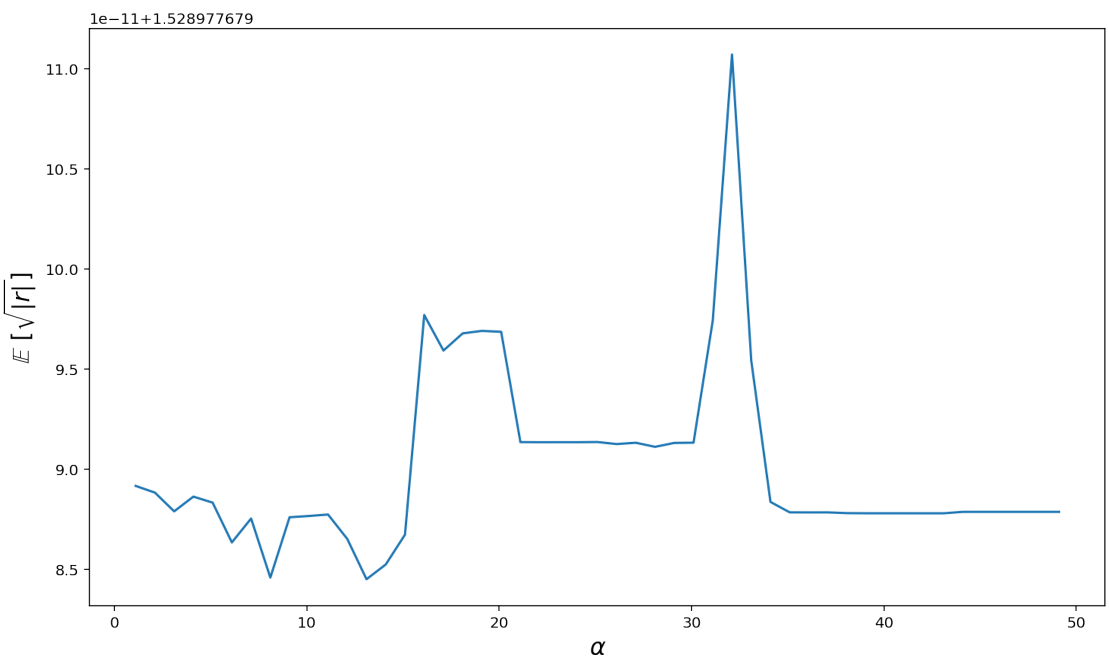
1.4 Partie 3 : Caractériser la dépendance de r en (σ, α)

- Représenter la moyenne empirique de $|r|^{1/2}$ en fonction de α.

[40]:
```
def m_r(a):
    m = []
    for x in a:
        m.append(np.mean(np.sqrt(np.abs(simulation(x)[0]))))
    return (m)

figure = plt.figure()#figsize=(17, 5))
ax1 = figure.add_subplot(111)
X = np.arange(1.1, 50, 1)
Y = m_r(X)
plt.plot(X, Y)
plt.ylabel('$\mathbb{E} \ [\sqrt{|r|}]$',fontsize=16)
plt.xlabel(r'$\alpha$',fontsize=16)
plt.plot()
```

[40]: []

[40]:



Interprétation :

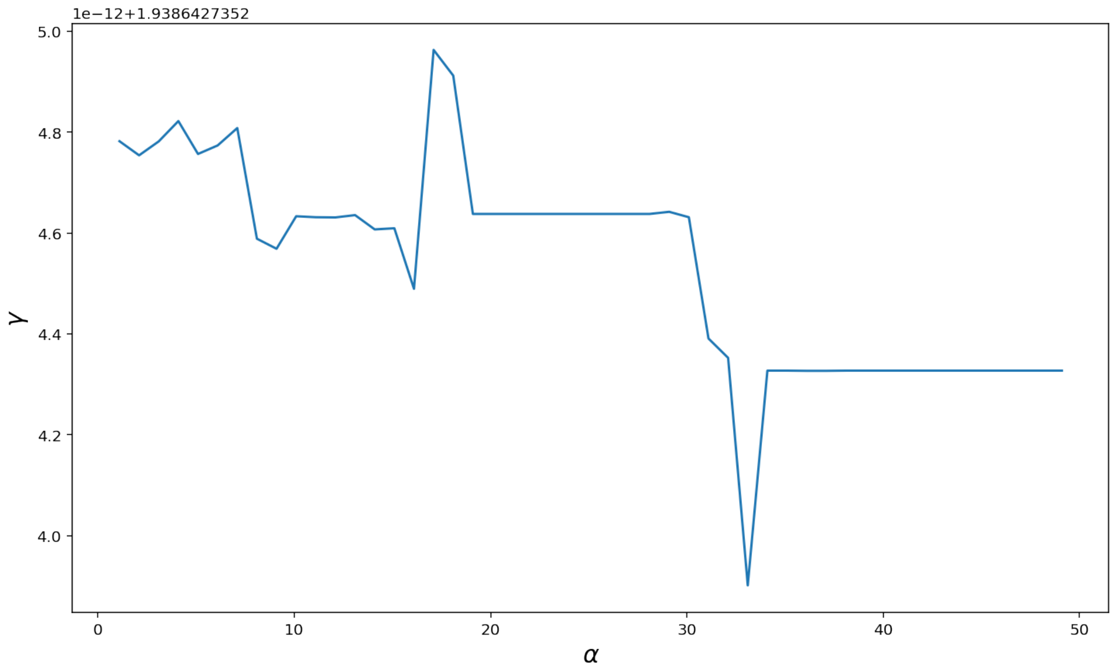
Les variations sont infimes, comme on pouvait s'y attendre $\mathbb{E}\ [|r|^2]$ ne dépend pas de α.
En effet, d'après le cours, α n'est pas présente dans la formule du rendement pour $\hat{a}$= α + $\frac{\epsilon_t}{r_{t-1}}$.

- Représenter la moyenne empirique de γ en fonction de α.

[41]:
```
def m_g(a):
    m = []
    for x in a:
        mypl = powerlaw.Fit(np.abs(simulation(x)[0][:1000]))  ##reduction du␣
↪nombre de points consideres
        m.append(mypl.alpha)
    return (m)


figure = plt.figure()#figsize=(17, 5))
ax1 = figure.add_subplot(111)
X = np.arange(1.1, 50, 1)
Y = m_g(X)
clear_output(wait=True)
plt.plot(X, Y)
plt.ylabel('$\gamma$',fontsize=16)
plt.xlabel(r'$\alpha$',fontsize=16)
plt.plot()
```

[41]: []

[41]:



Interprétation :

A σ fixé, γ ne varie pas en fonction de α.
La valeur de α semble avoir peu d'influence sur la modélisation choisie.

1.5 Bonus : la moyenne empirique de $|r|^{1/2}$ et γ en fonction de σ

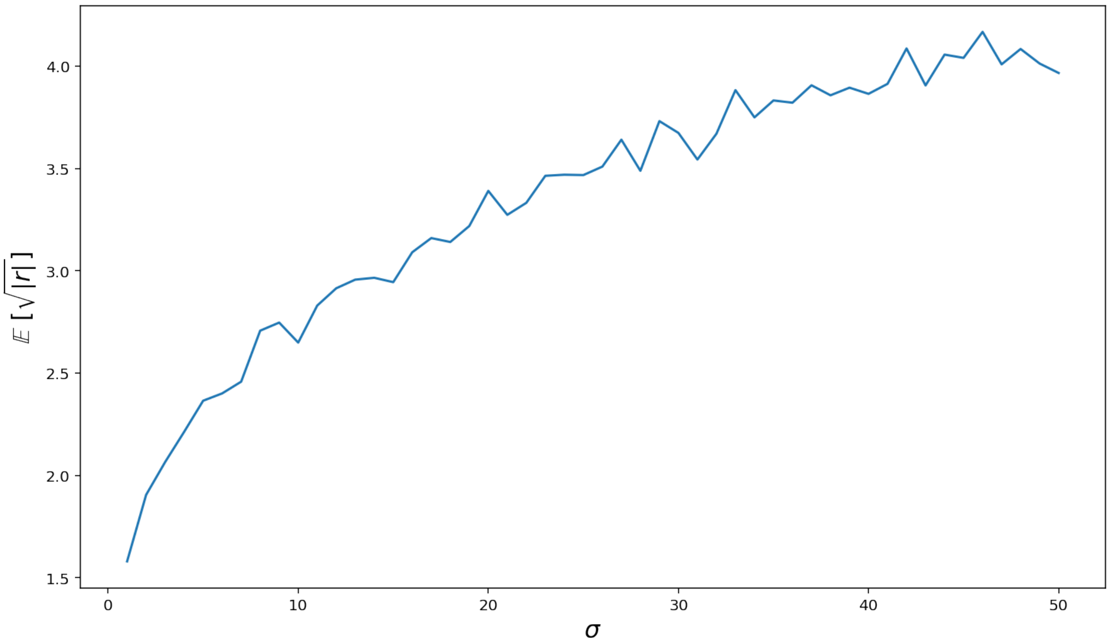
Comme cette partie n'était pas demandée, nous avons préféré réécrire une fonction *simulation* pour de ne pas complexifier les parties précédentes. Nous aurions pu simplement ajouter un argument à la fonction *simulation*.

[31]:
```
T=10000
a_0 = 2
r_0 = 1
S = np.arange(1, 51, 1)
E = [np.random.normal(scale=sqrt(s), size=T) for s in S]


def simulation_1(e):
    rendement = [r_0]
    alpha = [a_0]
    for t in range(1, T):
        rendement.append((a_0 - alpha[t - 1]) * rendement[t - 1] + e[t])
        alpha.append(rendement[t] / rendement[t - 1] + alpha[t - 1])
    return (rendement, alpha)
```

[32]:
```
def m_r_s(E):
    m = []
    for e in E:
        m.append(np.mean(np.sqrt(np.abs(simulation_1(e)[0]))))
    return (m)


figure = plt.figure()#figsize=(17, 5))
ax1 = figure.add_subplot(111)
Y = m_r_s(E)
plt.plot(S, Y)
plt.ylabel('$\mathbb{E} \ [\sqrt{|r|}]$',fontsize=16)
plt.xlabel(r'$\sigma$',fontsize=16)
plt.plot()
```

[32]: []

[32]:



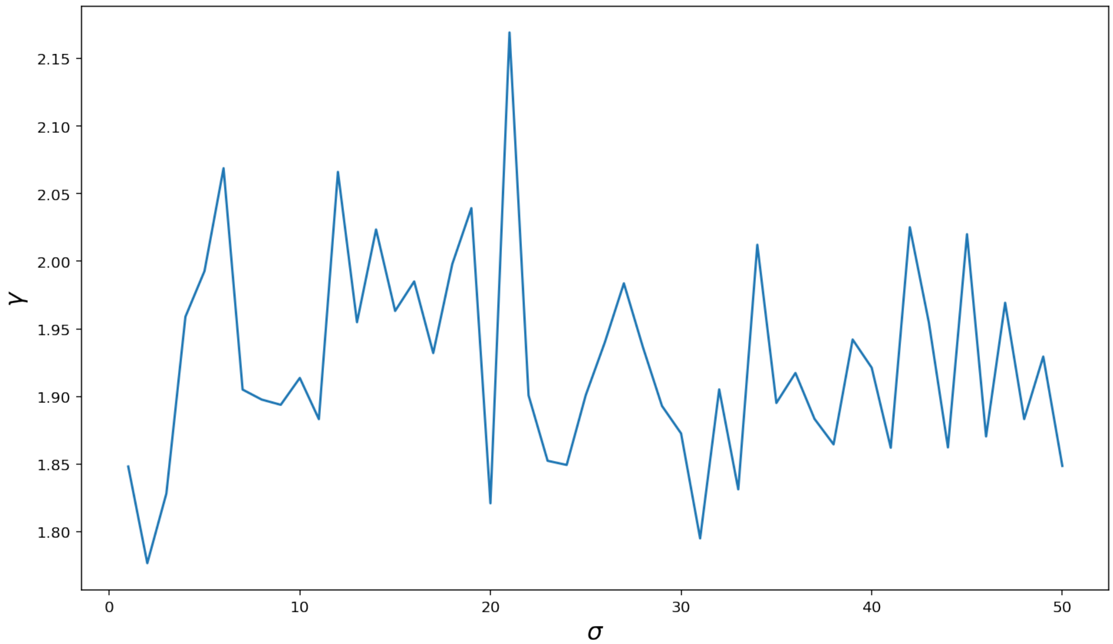
Interprétation :

L'évolution de $\mathbb{E}\ [|r|^{0.5}]$ est en $\sqrt{\sigma}$. La moyenne empirique de la racine des rendements augmente avec la volatilité ce qui semble cohérent. Cette observation est cohérente avec la formule de l'espérance.

[33]:
```
def m_g_s(E):
    m = []
    for e in E:
        mypl = powerlaw.Fit(np.abs(simulation_1(e)[0][:1000]))  # réduction du␣
 ↪nombre de points considérés pour diminuer le temps de calcul
        m.append(mypl.alpha)
    return (m)

figure = plt.figure() #figsize=(17, 5))
ax1 = figure.add_subplot(111)
Y = m_g_s(E)
clear_output(wait=True)
plt.plot(S, Y)
plt.ylabel('$\gamma$',fontsize=16)
plt.xlabel(r'$\sigma$',fontsize=16)
```

[33]: Text(0.5, 0, '$\\sigma$')

[33]:



Interprétation :

L'évolution de γ en fonction de σ ne saute pas aux yeux: γ évolue aléatoirement. Ce qui nous invite à nous demander de quoi dépend γ.

1.6 Conclusion

En conclusion, ce travail nous a permis de comprendre la dépendance entre les différents paramètres. α n'a pas d'influence sur le modèle. Le modèle choisi, bien qu'il soit simple, permet d'obtenir des résultats intéressants.

TP3

May 25, 2020

1 ST4 MDS - Données et Statistiques en Finance

1.1 TP 3: The Brock Hommes model

1.1.1 Martin Bridoux & Karim El Asmar

[3]:
```
import numpy as np
import matplotlib.pyplot as plt
from math import *
```

1.2 Partie 1 : Chaos avec trois stratégies triviales

Implémenter le modèle de Brock Hommes sous forme de fonction qui retourne le vecteur x_t

[2]:
```
def evaluate(H,x):
    return(np.array([f(x) for f in H]))

def BH(H,b,l):
    U = 0.5*np.ones(len(H))
    n = np.ones(len(H))
    z = np.zeros(len(H))
    x = [0.5]

    for t in range(1,T):
        n = np.exp(b*U)
        Z = np.sum(n)
        n /= Z
        #On suppose que f_h dépend que de x_t-1
        z = np.array([f(x) for f in H])
        x.append(1/(1+r) * np.sum(n*z))
        U = U*l + (1-l)*(x[t]-(1+r)*x[t-1])*(evaluate(H,x)-r*x[t-1])
    return x

def graphes(b,H,l):
    vecteur_x=BH(H,b,l)
    figure = plt.figure(figsize=(17,5))
```

```
    ax1=figure.add_subplot(121)
    ax2=figure.add_subplot(122)

    ax1.set_title("$x_t$ en fonction de t pour $\lambda="+str(l)+r"$,␣
↪$\beta="+str(b)+"$",fontsize=16)
    ax1.set_xlabel("t")
    ax1.set_ylabel("$x_t$")
    ax1.plot(vecteur_x[1:500])#on reduit le nombre de points pour bien voir les␣
↪creneaux

    ax2.set_title("$x_t$ en fonction de $x_{t-1}$ pour $\lambda="+str(l)+r"$,␣
↪$\beta="+str(b)+"$",fontsize=16)
    ax2.set_xlabel("$x_{t-1}$")
    ax2.set_ylabel("$x_t$")
    ax2.plot(vecteur_x[1:T],vecteur_x[0:T-1])
```

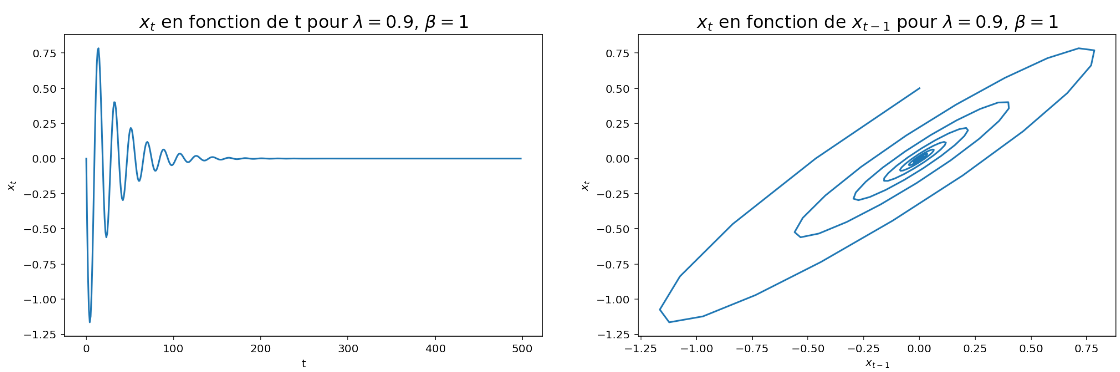
Définir trois stratégies : $f_0 = 0$, $f_1 = g$, $f_2 = -g$,
avec $r = 0,01$, $\lambda = 0,9$ et $g = 5$

[4]:
```
T=10000
r=0.01
l=0.9
g=3.75

f_0 = lambda x : 0
f_1 = lambda x : g
f_2 = lambda x : -g
H=[f_0,f_1,f_2]
```

1. Tracer x_t en fonction de t pour $\beta = 1$. Tracer également x_t en fonction de x_{t-1}.

[5]:
```
graphes(1,H,l)
```

[5]:



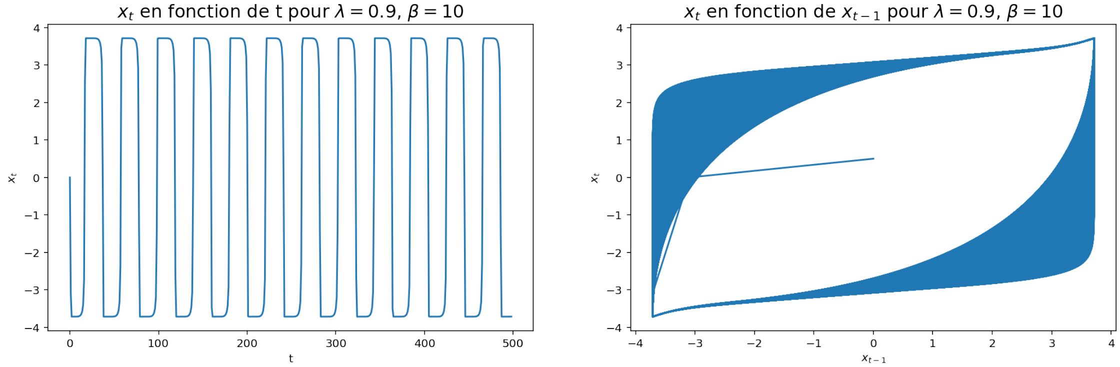
Interprétation:

Lorsqu'on modifie λ, l'amplitude des oscillations augmente, ce qui est cohérent puisque λ est un facteur d'oubli.
x_t est stable dans cette configuration, puisqu'on observe tant sur le tracé de x que sur le portrait de phase que $\lim_{t \to \infty} x_t = 0$.

2. Pareil pour β = 10. Comparer avec le cas $\beta = 1$. Pourquoi x = 0 devient-il instable ?

[6]:
```
graphes(10,H,l)
```

[6]:



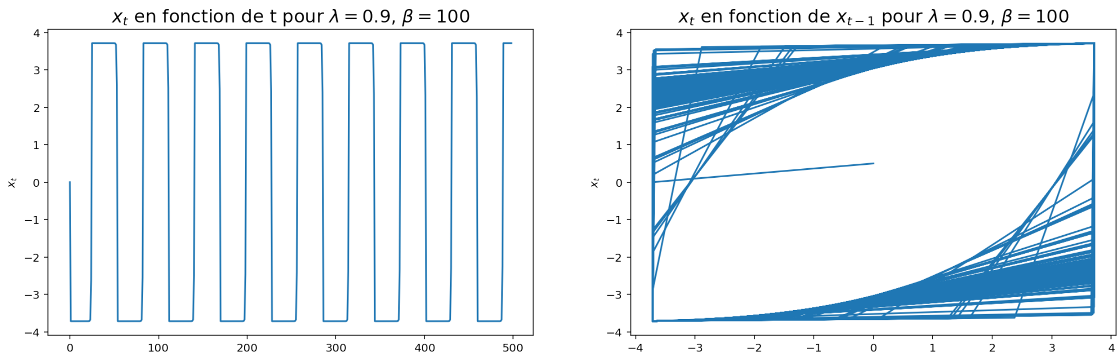
Interprétation:

Pour β=10, on remarque la présence de créneaux à valeurs dans $[-g, g]$ qui ne s'atténuent pas sur la graphe de gauche.
De plus, le diagramme de phase est circulaire: x_t est instable.

3. Pareil pour β = 100. Comparer avec le cas $\beta = 10$.

[7]:
```
graphes(100,H,l)
```

[7]:



Interprétation:

De manière générale pour β grand, on se fixe une stratégie. Donc pour β=100,

tout le monde utilise la même stratégie, on remarque que la période des oscillations est plus élevée que pour β=10 (mais nous ne savons pas trop expliquer physiquement ce dernier point).
x_t est évidemment instable.

1.3 Partie 2: Stratégies traditionnelles

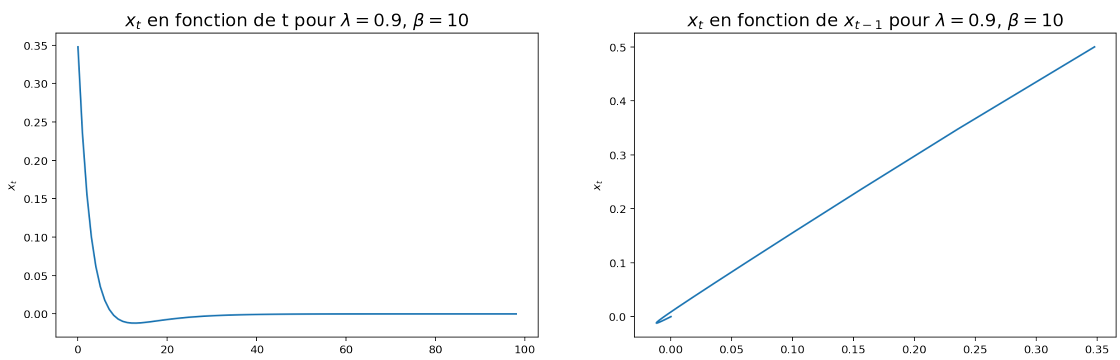
Définir 4 stratégies:
$f_0 = 0$
$f_1(x) = 0,9x + 0,2$
$f_2(x) = 0,9x - 0,2$
$f_4(x) = (1 + r)x$

[8]:
```
T=100
r=0.01
l=0.9

f_0 = lambda x : 0
f_1 = lambda x : 0.9*x[-1] + 0.2
f_2 = lambda x : 0.9*x[-1] - 0.2
f_4 = lambda x : (1+r)*x[-1]
H_1=[f_0,f_1,f_2,f_4]
```

1. Tracer x_t en fonction de t pour β = 10. Tracer également x_t en fonction de x_{t-1}.

[9]:
```
graphes(10,H_1,l)
```

[9]:



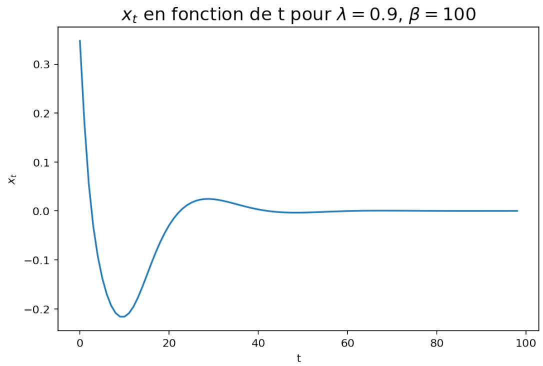
Interprétation:

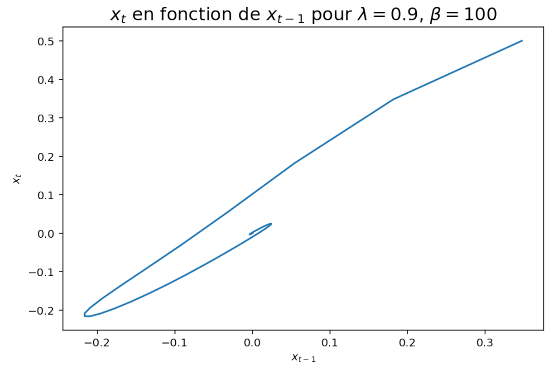
Dans cette partie l'influence des agents est linéaire, l'évolution de x_t n'est plus sous forme de créneaux mais varie continûment et tend à se stabiliser autour de la valeur 0, comme dans la première partie avec β=1.

2. Pareil pour β = 100.

[10]:
```
graphes(100,H_1,l)
```

[10]:





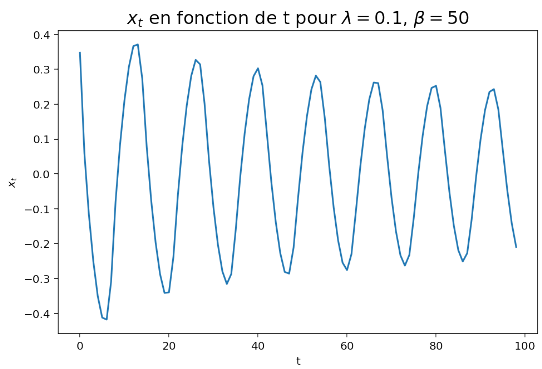
Interprétation:

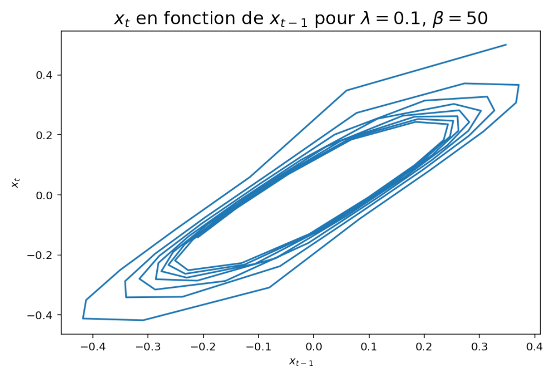
Quand β augmente, un régime transitoire apparait, cependant x_t reste stable.

3. Idem en prenant β = 50 et en augmentant λ. Est-ce que la stabilité disparait ?

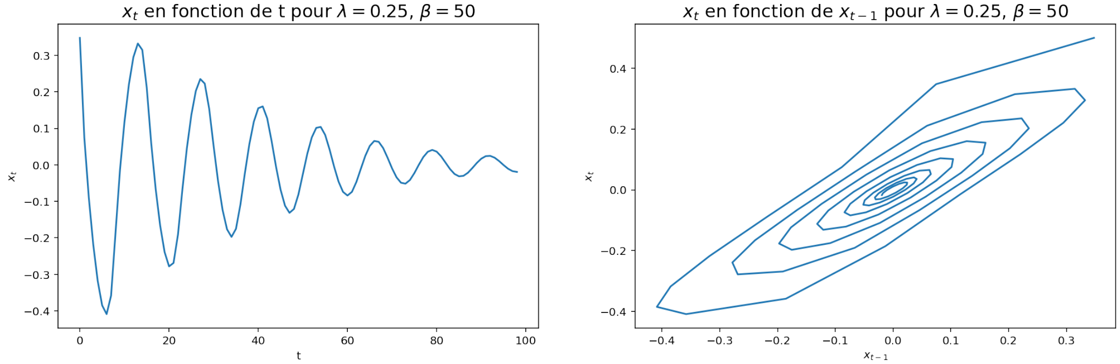
[11]:
```
L=np.linspace(0.1,1,7)

for l in L:
    graphes(50,H_1,l)
```

[11]:

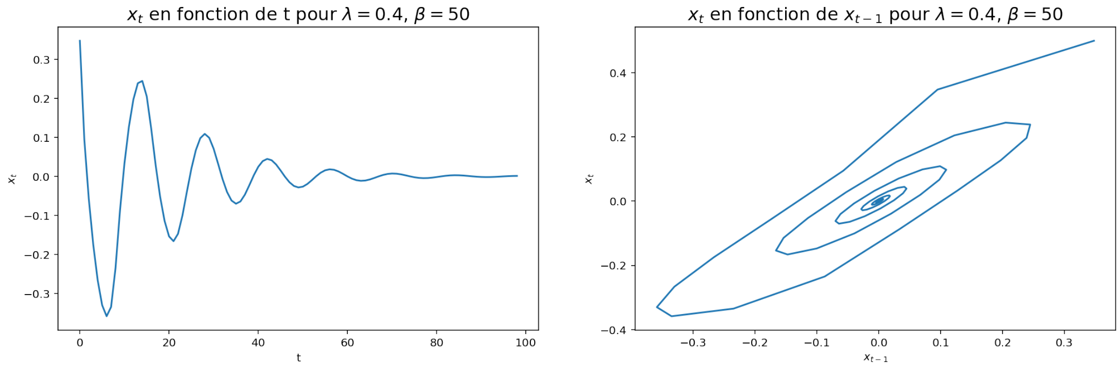




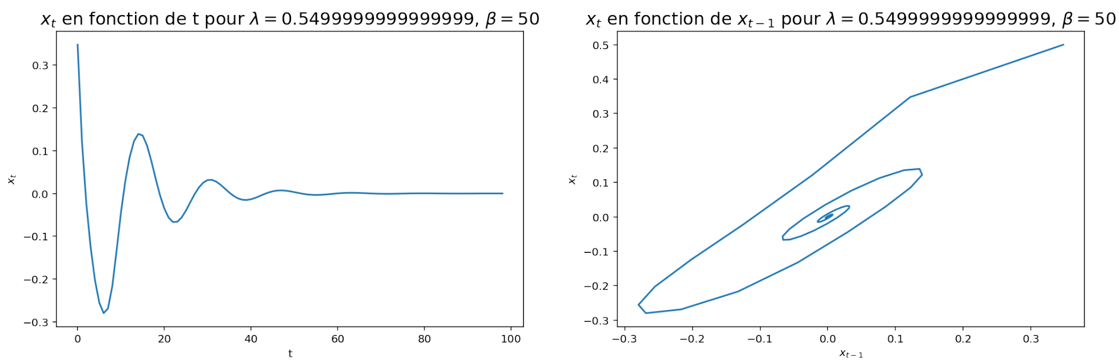
[11]:


x_t en fonction de t pour $\lambda = 0.25$, $\beta = 50$
x_t
t
x_t en fonction de x_{t-1} pour $\lambda = 0.25$, $\beta = 50$
x_t
x_{t-1}

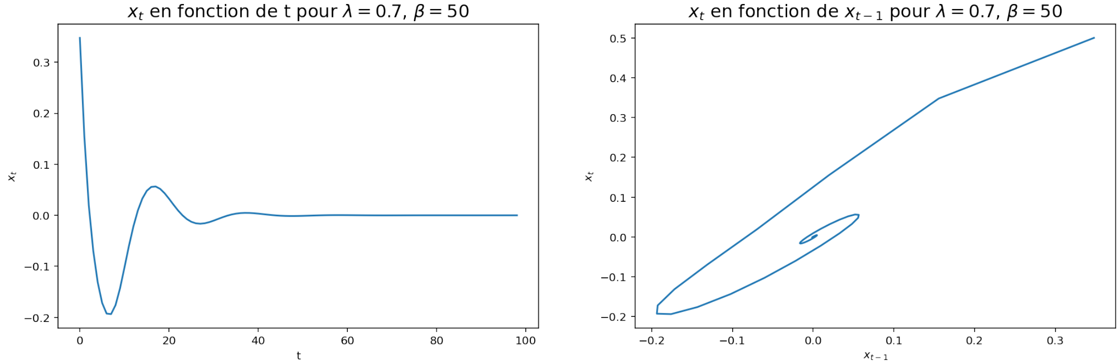
[11]:


x_t en fonction de t pour $\lambda = 0.4$, $\beta = 50$
x_t
t
x_t en fonction de x_{t-1} pour $\lambda = 0.4$, $\beta = 50$
x_t
x_{t-1}

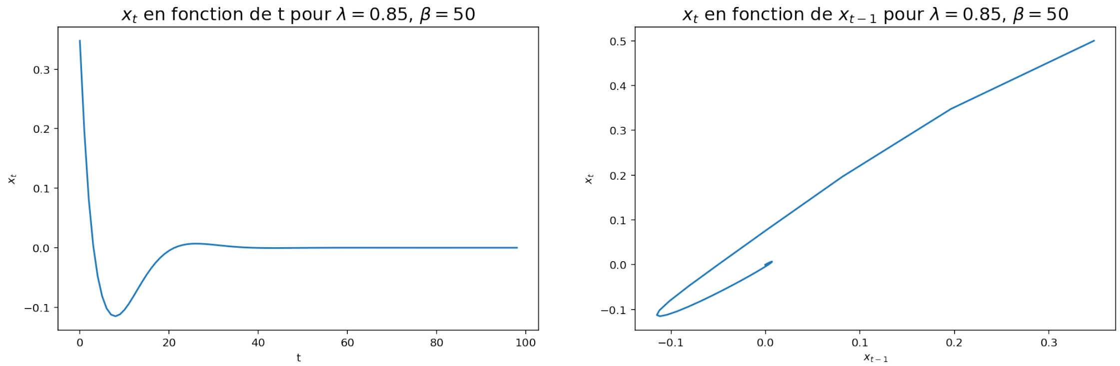
[11]:


x_t en fonction de t pour $\lambda = 0.5499999999999999$, $\beta = 50$
x_t
t
x_t en fonction de x_{t-1} pour $\lambda = 0.5499999999999999$, $\beta = 50$
x_t
x_{t-1}

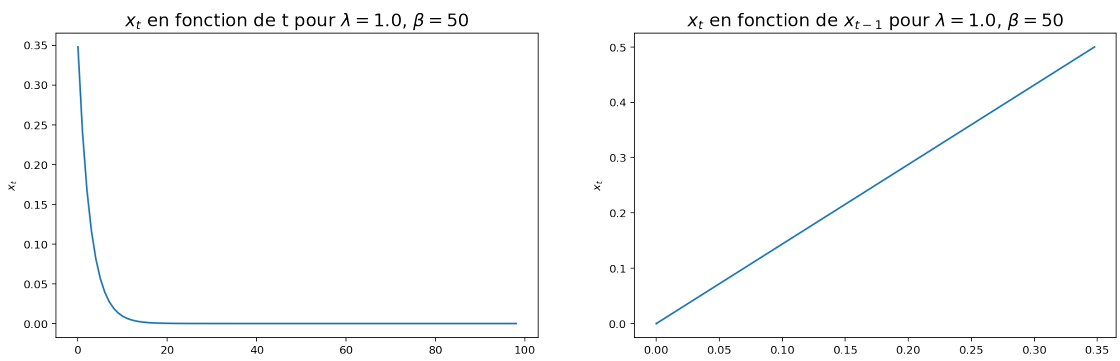
[11]:



[11]:



[11]:



Interprétation:

λ est le facteur d'oubli. Quand il tend vers 0, x_t est moins impacté par x_{t-1}, cela explique les oscillations faiblement amorties au voisinage de 0, typique d'un régime pseudo-périodique. On perd la stabilité.
Quand λ augmente, les oscillations observées tendent à disparaître au profit d'un régime apériodique et x_t devient stable (il tend vers 0).

1.4 Partie 3: Stratégies traditionnelles : Stratégies empiriques

Définir 5 stratégies telles que décrites dans le cours: ADA, WTR, STR, LAA, AA

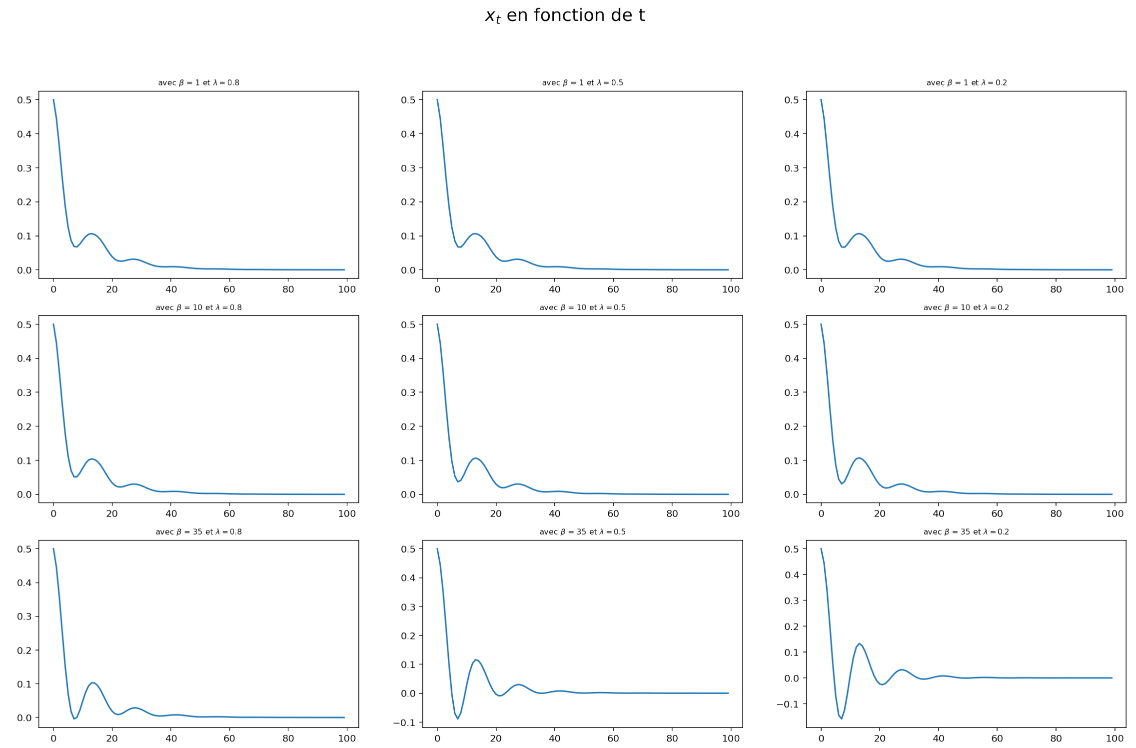
1. Tracer x_t en fonction de t pour plusieurs jeux de valeurs de paramètres. Quand est-ce que le modèle est stable/instable ?

[12]:
```
ADA_vect=[]
def ADA(x):
    if len(x)==1:
        ADA_vect.append(x[0])
        return ADA_vect[-1]
    ADA_vect.append(0.65*x[-1] + 0.35*ADA_vect[-1])
    return ADA_vect[-1]

def WTR(x):
    if len(x)==1:
        return x[-1]
    return x[-1]+0.4*(x[-1]-x[-2])

def STR(x):
    if len(x)==1:
        return x[-1]
    return x[-1]+1.3*(x[-1]-x[-2])

def LAA(x):
    if len(x)==1:
        return 0.5*(np.mean(x[-15:])+x[-1])
    return 0.5*(np.mean(x[-15:])+x[-1]) + (x[-1]-x[-2])

def AA(x):
    if len(x)==1:
        return 0.5*(x[-1])
    return 0.5*(x[-1]) + (x[-1]-x[-2])

fig = plt.figure(figsize=(20,12))
plt.suptitle('$x_{t}$ en fonction de t \n',fontsize=18)
ax1=fig.add_subplot(331)
ax2=fig.add_subplot(332)
ax3=fig.add_subplot(333)
ax4=fig.add_subplot(334)
ax5=fig.add_subplot(335)
ax6=fig.add_subplot(336)
ax7=fig.add_subplot(337)
ax8=fig.add_subplot(338)
```

```
ax9=fig.add_subplot(339)

vecteur_x=BH([ADA,WTR,STR,LAA,AA],1,0.8)
ax1.plot(vecteur_x)
ax1.set_title(r'avec $\beta$ = 1 et $\lambda = 0.8$',fontsize=8)


vecteur_x=BH([ADA,WTR,STR,LAA,AA],1,0.5)
ax2.plot(vecteur_x)
ax2.set_title(r'avec $\beta$ = 1 et $\lambda = 0.5$',fontsize=8)


vecteur_x=BH([ADA,WTR,STR,LAA,AA],1,0.2)
ax3.plot(vecteur_x)
ax3.set_title(r'avec $\beta$ = 1 et $\lambda = 0.2$',fontsize=8)

vecteur_x=BH([ADA,WTR,STR,LAA,AA],10,0.8)
ax4.plot(vecteur_x)
ax4.set_title(r'avec $\beta$ = 10 et $\lambda = 0.8$',fontsize=8)

vecteur_x=BH([ADA,WTR,STR,LAA,AA],10,0.5)
ax5.plot(vecteur_x)
ax5.set_title(r'avec $\beta$ = 10 et $\lambda = 0.5$',fontsize=8)

vecteur_x=BH([ADA,WTR,STR,LAA,AA],10,0.2)
ax6.plot(vecteur_x)
ax6.set_title(r'avec $\beta$ = 10 et $\lambda = 0.2$',fontsize=8)


vecteur_x=BH([ADA,WTR,STR,LAA,AA],35,0.8)
ax7.plot(vecteur_x)
ax7.set_title(r'avec $\beta$ = 35 et $\lambda = 0.8$',fontsize=8)


vecteur_x=BH([ADA,WTR,STR,LAA,AA],35,0.5)
ax8.plot(vecteur_x)
ax8.set_title(r'avec $\beta$ = 35 et $\lambda = 0.5$',fontsize=8)

vecteur_x=BH([ADA,WTR,STR,LAA,AA],35,0.2)
ax9.plot(vecteur_x)
ax9.set_title(r'avec $\beta$ = 35 et $\lambda = 0.2$',fontsize=8)
```

[12]:
```
Text(0.5, 1.0, 'avec $\\beta$ = 35 et $\\lambda = 0.2$')
```

[12]:

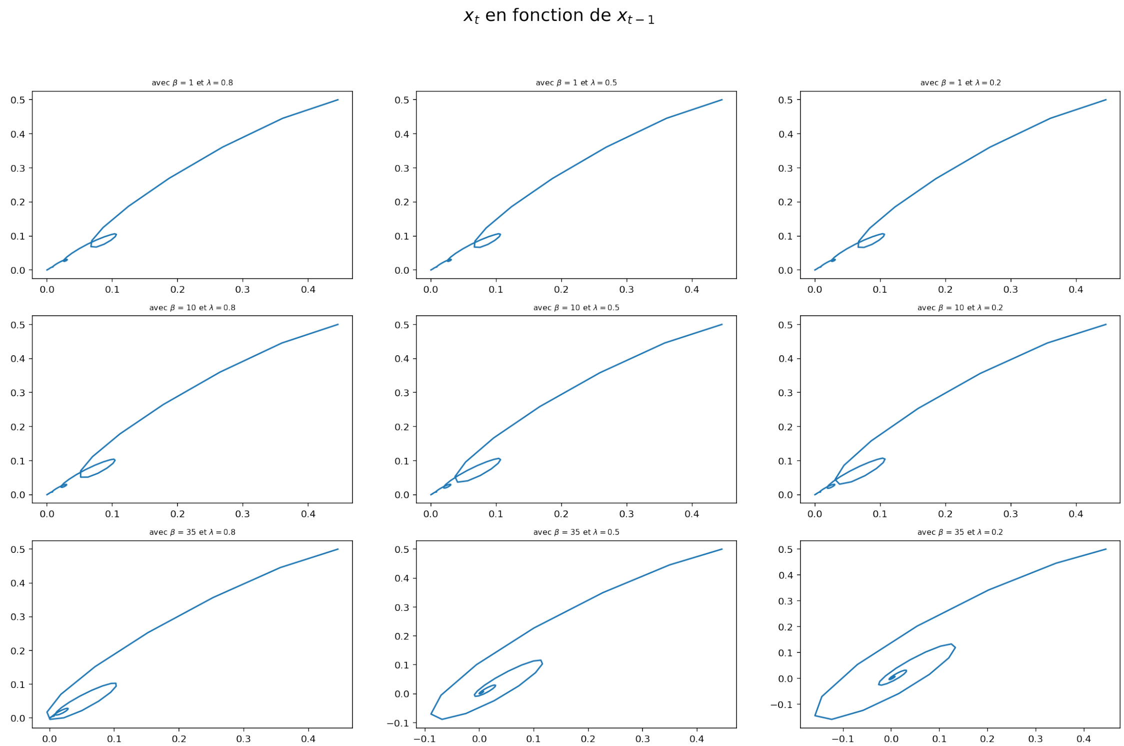


[14]:
```
fig = plt.figure(figsize=(20,12))
plt.suptitle('$x_{t}$ en fonction de $x_{t-1}$\n',fontsize=18)
ax1=fig.add_subplot(331)
ax2=fig.add_subplot(332)
ax3=fig.add_subplot(333)
ax4=fig.add_subplot(334)
ax5=fig.add_subplot(335)
ax6=fig.add_subplot(336)
ax7=fig.add_subplot(337)
ax8=fig.add_subplot(338)
ax9=fig.add_subplot(339)

vecteur_x=BH([ADA,WTR,STR,LAA,AA],1,0.8)
ax1.plot(vecteur_x[1:T],vecteur_x[0:T-1])
ax1.set_title(r'avec $\beta$ = 1 et $\lambda = 0.8$',fontsize=8)


vecteur_x=BH([ADA,WTR,STR,LAA,AA],1,0.5)
ax2.plot(vecteur_x[1:T],vecteur_x[0:T-1])
ax2.set_title(r'avec $\beta$ = 1 et $\lambda = 0.5$',fontsize=8)


vecteur_x=BH([ADA,WTR,STR,LAA,AA],1,0.2)
```

```
ax3.plot(vecteur_x[1:T],vecteur_x[0:T-1])
ax3.set_title(r'avec $\beta$ = 1 et $\lambda = 0.2$',fontsize=8)

vecteur_x=BH([ADA,WTR,STR,LAA,AA],10,0.8)
ax4.plot(vecteur_x[1:T],vecteur_x[0:T-1])
ax4.set_title(r'avec $\beta$ = 10 et $\lambda = 0.8$',fontsize=8)

vecteur_x=BH([ADA,WTR,STR,LAA,AA],10,0.5)
ax5.plot(vecteur_x[1:T],vecteur_x[0:T-1])
ax5.set_title(r'avec $\beta$ = 10 et $\lambda = 0.5$',fontsize=8)

vecteur_x=BH([ADA,WTR,STR,LAA,AA],10,0.2)
ax6.plot(vecteur_x[1:T],vecteur_x[0:T-1])
ax6.set_title(r'avec $\beta$ = 10 et $\lambda = 0.2$',fontsize=8)


vecteur_x=BH([ADA,WTR,STR,LAA,AA],35,0.8)
ax7.plot(vecteur_x[1:T],vecteur_x[0:T-1])
ax7.set_title(r'avec $\beta$ = 35 et $\lambda = 0.8$',fontsize=8)


vecteur_x=BH([ADA,WTR,STR,LAA,AA],35,0.5)
ax8.plot(vecteur_x[1:T],vecteur_x[0:T-1])
ax8.set_title(r'avec $\beta$ = 35 et $\lambda = 0.5$',fontsize=8)

vecteur_x=BH([ADA,WTR,STR,LAA,AA],35,0.2)
ax9.plot(vecteur_x[1:T],vecteur_x[0:T-1])
ax9.set_title(r'avec $\beta$ = 35 et $\lambda = 0.2$',fontsize=8)
```

[14]:
```
Text(0.5, 1.0, 'avec $\\beta$ = 35 et $\\lambda = 0.2$')
```

[14]:



Interprétation:

Indépendamment de la configuration considérée, x_t est stable. Quand β augmente, on se rapproche de la domination d'une stratégie, le regime transitoire est alors plus important : la convergence est plus lente. Quand le facteur d'oubli λ augmente le régime transitoire est plus long notamment quand β est grand.
A l'inverse, quand les stratégies sont mélangées (milieu homogène pour β=1), l'effet de λ est peu perceptible.

1.5 Conclusion

Nous tenions à vous remercier pour ces TPs qui nous ont permis d'avoir une vision plus claire de certains phénomènes.
