Institut National de la Recherche Agronomique

Centre de Recherches de Jouy-en-Josas

Unité de Mathématiques et Informatique Appliquées (MIA)

PROGRAMMATION MATLAB

du filtrage non linéaire par convolution de particules pour l'identification et l'estimation d'un système dynamique microbiologique

Version 2009_1

Caroline BIDOT

Jean-Pierre GAUCHI & Jean-Pierre VILA

Rapport technique : 2009-3

Avril 2009

Table des matières

Introduction

L'objectif de ce programme, en langage Matlab, est l'identification et l'estimation d'un système dynamique microbiologique complexe. La méthode programmée dans ce but est une méthode de filtrage non linéaire par convolution de particules complètement décrite dans les références [1, 2, 3] données en page 6. Le contexte microbiologique et le principe de la méthode sont succinctement évoqués en page 4.

C'est le programme de V. Rossi, doctorant (thèse soutenue en 2004) sous la direction de J.-P. Vila (INRA-SupAgro/UMR Analyse des Systèmes et Biométrie/Montpellier, France) qui a servi de base fondamentale au programme présent. Le travail de Rossi se retrouve ici dans la fonction « Fonction filpar_convolution2 » donnée en page 19 (on rappelle également une contribution de R. Choquet de l'Université Paris VI sur cette fonction de filtre).

En dehors de cette dernière fonction la programmation des autres fonctions de ce rapport technique a été assurée par C. Bidot (INRA/UR341: Mathématiques et Informatique Appliquées/Jouy-en-Josas).

J.-P. Vila a été le référent scientifique, et J.-P. Gauchi l'animateur du projet.

Nous remercions J.-C. Augustin (ENVA/Unité Microbiologie des Aliments – Sécurité et Qualité/Maisons-Alfort) qui a fourni un jeu de données réelles de microbiologie afin de valider ce programme, et qui continue maintenant la validation avec de nombreux autres jeux de données. L.Coroller, un autre microbiologiste (LUBEM : Lab. Universitaire de Biodiversité et d'Ecologie Microbienne, Quimper), validera également le programme avec d'autres jeux de données et d'autres modèles.

Ce rapport correspond à la première version dénommée Version 2009_1. Cette version met en œuvre seulement deux modèles de croissance primaire (Baranyi et Rosso [4, 5, 6]), et ne dispose pas d'interface graphique pour indiquer les choix de l'utilisateur. On ne prévoit pas sa diffusion. A court terme une deuxième version sera disponible, et diffusable, dans laquelle seront ajoutés :

- le calcul du facteur de Bayes pour la discrimination de deux modèles,
- l'optimisation des instants de prélèvements pour un coût expérimental donné, afin d'augmenter les vitesses de convergence des lois a posteriori estimées par filtrage,
- d'autres modèles,
- une interface graphique conviviale pour un utilisateur microbiologiste.

Enfin, il nous faut signaler que ce programme est la composante informatique de l'ARC INRIA-INRA (2009-2010) animée par P. Del Moral (INRIA Bordeaux) dont les membres sont J.-P. Vila, J.-P. Gauchi, C. Bidot, J.-C. Augustin et L. Coroller.

Contexte et principe de la méthode

Contexte

La complexité des systèmes microbiologiques de type alimentaire est à la fois de nature structurelle (écosystèmes impliquant plusieurs espèces bactériennes en interaction) et de nature fonctionnelle. Cette dernière résulte de la distribution de leur dynamique selon au moins trois niveaux hiérarchiques que l'on peut approximativement représenter par: a) un niveau externe, seul accessible aux mesures (comptages en milieu de culture après prélèvement et séries de dilutions), b) un niveau intermédiaire, correspondant aux dynamiques de croissances ou décroissances bactériennes proprement dites, non mesurables directement dans le substrat alimentaire considéré, mais a priori modélisables sous forme de modèles stochastiques dits primaires, c) un niveau plus profond, caractérisé par des variables cinétiques qui conditionnent les dynamiques précédentes, et elles-mêmes résultats d'interactions de différents facteurs biotiques et abiotiques (conditions de milieux telles le pH, la température, l'activité de l'eau, ..). Ces éléments cinétiques peuvent quelques fois être modélisables sous forme de modèles stochastiques dits secondaires.

Ces caractéristiques fonctionnelles rendent toute identification et modélisation prévisionnelle de ces dynamiques bactériennes par les approches classiques (e.g. moindres carrés non linéaires), particulièrement difficile sinon impossible. Des travaux récents [1, 2, 3] ont permis d'apporter une réponse méthodologique pertinente, aux deux problèmes de l'identification paramétrique de ces systèmes et de l'estimation des densités de probabilités prévisionnelles de l'évolution des concentrations bactériennes dans les substrats d'intérêt. Ils reposent sur la mise en œuvre d'une nouvelle technique de filtrage non linéaire particulaire, à noyau de convolution. Les fondements théoriques de la méthode sont publiés dans [1, 2, 3]. On se contentera d'indiquer ci-dessous quelques éléments.

Acquisition statistique des données

Les procédures actuelles d'acquisition des données sont séquentielles : à partir d'un tube primaire de solution, un prélèvement est effectué, dilué en cascade (dilutions en série de Fisher), suivi d'un étalement sur des boîtes de Petri. Après un certain temps les unités bactériennes formant colonies (UFC) sont dénombrées, sous l'hypothèse que chaque colonie

correspond à une seule bactérie déposée. La pratique actuelle des microbiologistes est de déduire l'effectif des bactéries présentes dans la solution initiale du tube primaire par simple extrapolation à partir des facteurs de dilutions successives en ignorant les erreurs d'échantillonnage (lors du prélèvement), de pipetage, de dilution, et de comptages sur les boîtes. Cette estimation déterministe très approximative est éminemment perfectible. Sous des hypothèses de répartitions spatiales à valider expérimentalement (distributions poissonniennes, distributions agrégatives) et de lois d'erreur de manipulation, nous avons effectué les caractérisations probabilistes de ces comptages (fonction des effectifs initiaux), prenant en compte des estimations des variances de toutes les erreurs successives.

Estimation de densités conditionnelles d'effectifs bactériens

Les systèmes dynamiques considérés ici s'apparentent aux systèmes à espace d'état non linéaires de forme générale formée d'une équation d'état autorégressive $x\{t+1\} = f(x\{t\},\alpha)$ (éventuellement à sortie vectorielle) et d'un modèle d'observation (éventuellement aussi à sortie vectorielle) de forme explicite $y = h(x\{t\},\beta)$ ou de loi de probabilité connue $y \sim L(x\{t\},\beta, .)$. Ces équations mettent en jeu un vecteur $\theta = (\alpha \quad \beta)^T$ de paramètres constants et inconnus, à estimer.

Dans l'approche par filtrage à convolution qui est la nôtre, on introduit une équation d'état supplémentaire sur le vecteur de paramètres : $\theta\{t+1\} = \theta\{t\}$, pour associer l'estimation des densités conditionnelles des paramètres θ à celle des variables d'état $x\{t\}$. A partir de densités a priori à l'instant t=0 pour le vecteur des variables et pour le vecteur des paramètres, ces estimations à l'instant t s'obtiennent par les formules données dans [1, 2, 3].

Conclusion

Cette approche d'identification par filtrage particulaire d'un système dynamique microbiologique, constitue une démarche innovante pour la communauté des microbiologistes. Elle repose sur une prise en compte exhaustive de toutes les erreurs de manipulations et de comptage et permet une estimation rigoureuse des paramètres des modèles (ce qui n'est pas le cas avec les approches actuelles par pseudo-moindres carrés double passe). Elle devrait permettre d'améliorer de façon sensible les démarches de microbiologie prévisionnelle particulièrement utile pour une maîtrise des risques alimentaires.

Bibliographie

[1] Rossi, V., Vila, J.P. (2003) Filtrage non linéaire en temps discret par convolution de particles. Actes des XXXVèmes Journées de Statistique, 823-826.

[2] Rossi, V., Vila, J.P. (2005) Approche non paramétrique du filtrage de système non linéaire à temps discret et à paramètres inconnus. C.R. Acad. Sci. Paris. Ser I 340, 759-764.

[3] Rossi, V., Vila, J.P. (2006) Nonlinear filtering in discrete time : a particle convolution approach. Inst. Stat. Univ. Paris, 3, 71-102.

[4] Baranyi, J., Roberts, T.A. (1994). A dynamic approach to predicting bacterial growth in food. International Journal of Food Microbiology, 23,277-294.

[5] Baranyi, J., Roberts, T.A. (1995). Mathematics of predictive food microbiology International Journal of Food Microbiology, 26, 199-218.

[6] Rosso, L., Lobry, J.R., Flandrois, J.P. (1993) An unexpected correlation between cardinal temperatures of microbial growth highlighted by a new model. J. of Theoretical Biology, 162, 447-463.

Listings sources

Procédure MainmanipJCA

```
%% MainmanipJCA :
% Programme/script pour traiter donnees d'un fichier excel
resultant d'examens microbiologiques
% Estimation des parametres d'un modele de croissance (Baranyi
ou Rosso)
% par l'utilisation d'un filtre particulaire par convolution

%% Tout d'abord on fait place nette
close all
clear all

%% Declaration de variables globales
global CVpesee CVpipetee CVdiluant
global mumaxmin mumaxmax lambdamin lambdamax N0min N0max
Nmaxmin Nmaxmax

%% Choix de la graine de l'alea
%%%%%****%%%%%%%%%%****%%%%%
graineAlea=sum(100*clock); % cette valeur est modifiable par
l'utilisateur
%%%%%****%%%%%%%%%%****%%%%%
try

RandStream.setDefaultStream(RandStream('mt19937ar','seed',grai
neAlea)); % version 2008b matlab
catch
    rand('state', graineAlea); % versions anterieures
end

%% Recuperation des donnees
% indiquer ci-dessous le fichier excel a ouvrir en precisant
le chemin
% d'acces eventuellement
%%%%%****%%%%%%%%%%****%%%%%
releveobs=xlsread('ex_cinetique_MIA.xls');
%%%%%****%%%%%%%%%%****%%%%%

% temps d'observations - 1ere colonne du fichier
tobs=releveobs(:,1);
[tobs,I,J]=unique(tobs);
% nombre d'observations
nbobs=length(tobs);
% nombre de repetitions
nbrepet=length(J)/nbobs;
disp(['On dispose de ' num2str(nbobs) ' temps de mesures et de
' num2str(nbrepet) ' repetitions de chaque mesure.']);
```

```
% le reste (parametres et donnees)
m=releveobs(:,2); % masse prelevee - 2nde colonne du fichier
a=releveobs(:,3); % facteur de dilution initial - 3eme colonne
du fichier
n=releveobs(:,4); % nombre de dilutions en tube - 4eme colonne
du fichier
F=releveobs(:,5); % facteur de dilution total - 5eme colonne
du fichier
v=releveobs(:,6); % volume ensemence - 6eme colonne du fichier
nb=releveobs(:,7); % nb d'ufc comptees - 7eme colonne du
fichier
yestime=releveobs(:,8:9); % estimations nb total (nb et log) -
8 & 9eme colonnes du fichier

% on rearrange les vecteurs en tableaux de taille nbrepet x
nbobs
% nb d'ufc comptees
yobs=reshape(nb,nbrepet,nbobs);%zeros(nbrepet,length(I));
% parametres
m2=reshape(m,nbrepet,nbobs);
a2=reshape(a,nbrepet,nbobs);
n2=reshape(n,nbrepet,nbobs);
F2=reshape(F,nbrepet,nbobs);
v2=reshape(v,nbrepet,nbobs);
% estimations nb total (nb et log)
Nbestime=reshape(yestime(:,1),nbrepet,nbobs);
logNbestime=reshape(yestime(:,2),nbrepet,nbobs);

% creation du tableau contenant les parametres de la fonction
d'observation
nbparam=5; % nombre de parametres
paramobs=zeros(nbrepet,nbparam,nbobs); % format obligatoire
pour passage dans la fonction de filtre
paramobs(:,1,:)=m2; % masse prelevee
paramobs(:,2,:)=a2; % facteur de dilution initial
paramobs(:,3,:)=n2; % nombre de dilutions en tube
paramobs(:,4,:)=F2; % facteur de dilution total
paramobs(:,5,:)=v2; % volume ensemence

% on efface les variables intermediaires qui ne serviront plus
clear releveobs m a n F v nb m2 a2 n2 F2 v2 yestime

%% Initialisation des valeurs des coefficients de variation
maximum
%%%%%****%%%%%%%%%****%%%%%
% pour la pesee
CVpesee=0.025;
% pour la pipetee
CVpipetee=0.0025;
% pour le diluant
CVdiluant=0.01;
```

```
%%%%%****%%%%%%%%%%****%%%%%

%% Initialisation des valeurs min et max des parametres :
%%%%%****%%%%%%%%%%****%%%%%
% mumax
mumaxmin=0.01;
mumaxmax=2;
% lambda
lambdamin=20;
lambdamax=50;
% N0
N0min=100;
N0max=400;
% Nmax
Nmaxmin=1e8;
Nmaxmax=1e9;
%%%%%****%%%%%%%%%%****%%%%%

%% Choix du modele
%%%%%****%%%%%%%%%%****%%%%%
% actuellement Baranyi ou Rosso (choisir l'une des 2 lignes)
%modelepop=@Baranyi;
modelepop=@Rosso;

% et du pas de temps pour la dynamique (en heures)
% doit etre < min(tobs(i+1)-tobs(i))
deltaT=24;
%%%%%****%%%%%%%%%%****%%%%%

%% Le filtre

% nombre de particules pour l'utilisation du filtre
%%%%%****%%%%%%%%%%****%%%%%
nbparticules=1e5;
%%%%%****%%%%%%%%%%****%%%%%

% affichage (1) ou non (0) des histogrammes au cours du calcul
%%%%%****%%%%%%%%%%****%%%%%
tracehistogramme=1;
%%%%%****%%%%%%%%%%****%%%%%

% calcul :
tic
[Moyenne,ET,ICInf,ICSup]=filpar_convolution2(yobs,tobs,paramob
s,[0,tobs(end)],deltaT,modelepop,@manipJCA,nbparticules,@prior
B,[3,3,2,2,0],tracehistogramme);
toc % affichage du temps de calcul

%% Recuperation/exploitation des resultats et affichage
graphique
```

```
disp('Estimations obtenues pour les parametres, dans l''ordre
[mumax;lambda;N0;Nmax] :')
paramfin=[Moyenne(1:4,end);0];
disp(num2str(paramfin(1:4)'))


abscisses=1:nbobs;

nbfig=4*tracehistogramme;

if (tracehistogramme)
    figure(1)
    title('Distribution des estimations de mumax')
    figure(2)
    title('Distribution des estimations de lambda')
    figure(3)
    title('Distribution des estimations de N0')
    figure(4)
    title('Distribution des estimations de Nmax')
end

figure(nbfig+1)
plot(abscisses,Moyenne(1,:),'-',abscisses,ICInf(1,:),'--
',abscisses,ICSup(1,:),'-.')
title('Evolution des estimations de mumax')
axis([1,nbobs,-Inf,Inf])
xlabel('Observations')
ylabel([num2str(nbparticules) ' particules'])
legend('Moyenne','Borne inf IC a 95%','Borne sup IC a 95%',0)

figure(nbfig+2)
plot(abscisses,Moyenne(2,:),'-',abscisses,ICInf(2,:),'--
',abscisses,ICSup(2,:),'-.')
title('Evolution des estimations de lambda')
axis([1,nbobs,-Inf,Inf])
xlabel('Observations')
ylabel([num2str(nbparticules) ' particules'])
legend('Moyenne','Borne inf IC a 95%','Borne sup IC a 95%',0)

figure(nbfig+3)
plot(abscisses,Moyenne(3,:),'-',abscisses,ICInf(3,:),'--
',abscisses,ICSup(3,:),'-.')
title('Evolution des estimations de N0')
axis([1,nbobs,-Inf,Inf])
xlabel('Observations')
ylabel([num2str(nbparticules) ' particules'])
legend('Moyenne','Borne inf IC a 95%','Borne sup IC a 95%',0)

figure(nbfig+4)
plot(abscisses,Moyenne(4,:),'-',abscisses,ICInf(4,:),'--
',abscisses,ICSup(4,:),'-.')
```

```
title('Evolution des estimations de Nmax')
axis([1,nbobs,-Inf,Inf])
xlabel('Observations')
ylabel([num2str(nbparticules) ' particules'])
legend('Moyenne','Borne inf IC a 95%','Borne sup IC a 95%',0)

% Estimation de la dynamique de la pop avec les parametres du
modele
% obtenus
temps=0:deltaT:(tobs(end)+10);
pop=zeros(1,length(temps));
for i=1:length(temps)
    t=temps(i);
    temp=modelepop(paramfin,t,0);
    pop(i)=temp(5);
end
figure(nbfig+5)
plot(temps,log10(pop),'k')
hold on
plot(tobs,logNbestime,'o')
hold off
title('concentration')
legend('estimation','donnees',0)
ylabel('echelle log10')
xlabel('Temps')

figure(nbfig+6)
plot(temps,pop,'k')
hold on
plot(tobs,Nbestime,'o')
hold off
title('concentration')
legend('estimation','donnees',0)
%ylabel(' ')
xlabel('Temps')
```

Fonction Baranyi

```
function Xtp1=Baranyi(Xt,t,typebruit)
%% Fonction qui applique le modele de croissance de Baranyi
% function Xtp1=Baranyi(Xt,t,typebruit)
% avec Xt=(mumax; lambda; N0; Nmax; effectif) et au temps t
% et eventuellement ajout de bruit (typebruit=0/1)
% le resultat est sous la forme Xtp1=(mumax; lambda; N0; Nmax;
effectif)
% Xt peut avoir plusieurs colonnes (de taille 5xn)

% Recuperation des parametres
mumax=Xt(1,:);
lambda=Xt(2,:);
N0=Xt(3,:);
Nmax=Xt(4,:);
Xtp1=Xt;

% Calculs intermediaires
%A=t+log(exp(-mumax.*t)+exp(-mumax.*lambda)-exp(-mumax.*t-
mumax.*lambda))./mumax;
expmumaxA=1+exp(mumax.*(t-lambda))-exp(-mumax.*lambda);
B=1+(expmumaxA-1).*N0./Nmax;

% Calcul nouvel effectif
Xtp1(5,:)=N0.*expmumaxA./B;

% on efface les variables temporaires pour gain temps de
calcul
clear mumax lambda N0 Nmax expmumaxA B

% ajouter le bruit
if typebruit==1

Xtp1(5,:)=0.99.*Xtp1(5,:)+poissrnd(Xtp1(5,:)*0.01);%Xtp1(5,:)+
poissrnd(Xtp1(5,:)*0.01)-(Xtp1(5,:)*0.01);
end
```

Fonction Rosso

```
function Xtp1=Rosso(Xt,t,typebruit)
%% Fonction qui applique le modele de croissance de Rosso
% function Xtp1=Rosso(Xt,t,typebruit)
% avec Xt=(mumax; lambda; N0; Nmax; effectif) et au temps t
% et eventuellement ajout de bruit (typebruit=0/1)
% le resultat est sous la forme Xtp1=(mumax; lambda; N0; Nmax;
effectif)
% Xt peut avoir plusieurs colonnes (de taille 5xn)

% Recuperation des parametres
mumax=Xt(1,:);
lambda=Xt(2,:);
N0=Xt(3,:);
Nmax=Xt(4,:);
Xtp1=Xt;

% Calcul nouvel effectif modele de Rosso
if (t<=lambda)
    Xtp1(5,:)=N0;
else
    Xtp1(5,:)=Nmax./(1+((Nmax./N0)-1).*exp(-mumax.*(t-
lambda)));
end

% ajouter le bruit
if typebruit==1
    Xtp1(5,:)=0.99.*Xtp1(5,:)+poissrnd(Xtp1(5,:)*0.01);%-
(Xtp1(5,:)*0.01);
end

% on efface les variables temporaires pour gain temps de
calcul
clear mumax lambda N0 Nmax
```

Fonction PriorB

```
function [Xt,Xmin,Xmax]=priorB(n)
%% Pour definir l'a priori sur les parametres
% function [Xt,Xmin,Xmax]=priorB(n)
% fonctions loi a priori (uniformes) + definition du support
% n nombre de particules
% cette fonction a besoin de la declaration globale des min et
des max des
% parametres mumax lambda N0 et Nmax

global mumaxmin mumaxmax lambdamin lambdamax N0min N0max
Nmaxmin Nmaxmax

pmumax=unifrnd(mumaxmin,mumaxmax,1,n);
plambda=unifrnd(lambdamin,lambdamax,1,n);
pN0=unifrnd(N0min,N0max,1,n);
pNmax=unifrnd(Nmaxmin,Nmaxmax,1,n);
pNo=pN0;

if (nargout==3)
    Xmin=zeros(5,n);
    Xmin(1,:)=mumaxmin;
    Xmin(2,:)=lambdamin;
    Xmin(3,:)=N0min;
    Xmin(4,:)=Nmaxmin;
    %Xmin=repmat([mumaxmin;lambdamin;N0min;Nmaxmin;0],1,n);

%Xmax=repmat([mumaxmax;lambdamax;N0max;Nmaxmax;Nmaxmax],1,n);
    Xmax=Nmaxmax.*ones(5,n);
    Xmax(1,:)=mumaxmax;
    Xmax(2,:)=lambdamax;
    Xmax(3,:)=N0max;
end

Xt=[pmumax;plambda;pN0;pNmax;pNo];% vecteur etat pour
initialiser calcul du filtre

% on efface les variables temporaires pour gain temps de
calcul
clear pmumax plambda pN0 pNmax pNo
```

Fonction manipJCA

```
function Y=manipJCA(X,parametres)
%% Fonction qui pour un etat X ressort la valeur de la sortie
observee Y
%   function Y=manipJCA(X,parametres)
% X(end,:) : nb de bacteries par gramme dans le tube primaire
- derniere ligne de X...
%   qui peut etre de taille 5xn dans le cas Baranyi/Rosso (4
parametres + effectif)
% parametres : matrice de taille nbRepetx5, les colonnes
representent les
%   parametres d'observation : masse prelevee, facteur de
dilution initial,
%   nombre de dilutions en tube, facteur de dilution total et
volume ensemence
% Y : nb de bacteries comptees dans les boites de petri
(taille :nbRepetxn)
% cette fonction a besoin de la declaration globale des
coefficients de
%   variation max de pesee, de pipetee et de diluant

global CVpesee CVpipetee CVdiluant
% ces variables globales sont les coefficients de variation
max

%% Initialisation
n=length(X(end,:)); % nb de particules
nbRepet=size(parametres,1); % nb de repetitions
masseprelevee=parametres(:,1); % m masse prelevee
factDilInitial=parametres(:,2); % a facteur de dilution
initial
mctea=masseprelevee.*(1-factDilInitial)./factDilInitial; %
m*(1-a)/a
nbDilutions=parametres(:,3); % n nombre de dilutions en tube
factDilTotal=parametres(:,4); % F facteur de dilution total
cteF=(1-factDilTotal)./(10.*factDilTotal); % (1-F)/10F
Ve=parametres(:,5); % v volume ensemence

NbactPetri=zeros(nbRepet,n);

for rep=1:nbRepet

    % on a X bacteries par gramme
    NbactPetri(rep,:)=X(end,:);

    %% Etape 1 : Prelevement pour suspension mere
```

```
    % on prend m grammes qu'on "dilue" avec un facteur a

massev=normrnd(masseprelevee(rep),masseprelevee(rep).*CVpesee,
1,n); % = m
    massetot=normrnd(mctea(rep),mctea(rep).*CVpesee,1,n); % =
M = m*(1-a)/a
    massetot=massetot+massev; % = M+m = m/a

    % nb de bacteries transferees dans m grammes N1 = m * X
    NbactPetri(rep,:)=poissrnd(NbactPetri(rep,:).*massev);
    % dans le volume m+M on a toujours N1 bacteries
    clear massev

    if (nbDilutions>0)
        %% Etape 2 : Dilutions en tube et en cascade
        % on dilue n fois pour un facteur de dilution total F
        % on a un volume pipete de equivalent a 10F/(1-F) =
1/cteF
        Vp=normrnd(1./cteF(rep),CVpipetee./cteF(rep),1,n);
        % pour un volume de diluant de 10ml
        Vdil=normrnd(10,10.*CVdiluant,1,n);

        % nb de bacteries transferees N2 = N1 * Vp/(m+M) =
N1.a.10F/m(1-F)

NbactPetri(rep,:)=poissrnd(NbactPetri(rep,:).*Vp./massetot);
        % petit ajustement
        NbactPetri(rep,NbactPetri(rep,:)==0)=1;

        %% Etape 3 : Volume preleve pour ensemencement boites
Petri
        % on pipete Ve ml dans le volume Vp+Vdil
        Vev=normrnd(Ve(rep),Ve(rep).*CVpipetee,1,n);

        % nb de bacteries transferees N3 = N2 * Ve/(Vp+Vdil) =
N2.Ve.(1-F)/10

NbactPetri(rep,:)=poissrnd(NbactPetri(rep,:).*Vev./(Vp+Vdil));
        % petit ajustement
        NbactPetri(rep,NbactPetri(rep,:)==0)=1;
        clear Vp Vdil

    else % if (nbDilutions>0)
        %% Pas de dilution -> on saute l'etape 2
        %% Etape 3 : Volume preleve pour ensemencement boites
Petri
        % on pipete Ve ml dans le volume (m+M)
        Vev=normrnd(Ve(rep),Ve(rep).*CVpipetee,1,n);

        % nb de bacteries transferees N3 = N1 * Ve/(m+M)
```

```
NbactPetri(rep,:)=poissrnd(NbactPetri(rep,:).*Vev./massetot);
        % petit ajustement
        NbactPetri(rep,NbactPetri(rep,:)==0)=1;
    end % if (nbDilutions>0)
    clear massetot Vev

    %% Etape 4 : Comptage
    % moyenne et ecart-type pour la log-normale :
    m2=NbactPetri(rep,:).^2;
    sigmaCompt=NbactPetri(rep,:)./50;
    v=sigmaCompt.^2;
    mu = log((m2)./sqrt(v+m2));
    sigma = sqrt(log(v./(m2)+1));

    % nb de bacteries comptees
    NbactPetri(rep,:)=round(lognrnd(mu,sigma));
    clear mu sigma m2 sigmaCompt v
    % petit ajustement
    NbactPetri(rep,isnan(NbactPetri(rep,:)))=1;
    NbactPetri(rep,NbactPetri(rep,:)==0)=1;

end % for rep=1:nbRepet

Y=NbactPetri;

% on efface les variables temporaires pour gain temps de
calcul
clear NbactPetri
```

Fonction filpar_convolution2

basée sur le programme de V. Rossi (doctorant sous la direction de JP Vila, thèse 2004) et R. Choquet (Université Paris VI.

```
function
[Moyenne,ET,ICInf,ICSup]=filpar_convolution2(yobs,tobs,paramob
s,SupportT,deltaT,fetat,hobsbruit,n,pi0,typebruitetat,affichag
e)
%=======================================================
% PURPOSE Mise en oeuvre du filtre particulaire avec
convolution
%=======================================================
% INPUT yobs         : variable d'observation
%       tobs :  temps d'observations
%       paramobs : parametres lies aux observations, dim
:(nbrepet,nbparam,nbobs)
%       SupportT : [ t initial , t final] intervalle de temps
considere
%       deltaT : pas de temps pour le calcul des modeles
%           doit etre < min(tobs(i+1)-tobs(i))
%       fetat(fonction)         : Xt+1=f(Xt,bruit)
%       hobsbruit(fonctions)  : Yt+1=hobs(Xt,bruit,paramobs)
%       n                       : nb de particules
%       pi0(fonction)           : loi initiale de l'etat
(prior)
%=======================================================
% OUTPUT Moyenne : moyenne des etats a tout les instants
(nbligne=nb var obs, nbcol= durï¿½e(T))
%        ET : ecarts types des etats a tout les instants
(nbligne=nb var obs, nbcol= durï¿½e(T))
%        ICInf ICSup : I-C a 95% des etats a tout les instants
(nbligne=nb var obs, nbcol= durï¿½e(T))
%=======================================================
% last modified RC & VR 15.11.05 -> CB 17.03.09
%=======================================================

%%
%%%%%% Initialisation %%%%%%%%%%
% test sur le nb d'arguments passes a la fonction
if nargin==10
    affichage=false;
end
% nb d'occasions
```

```
nbobs=length(tobs);
epsilon=1e-8; % precision pour trouver a quel tps
d'observation on est
nbtraitement=3; % nb de traitements possibles pour l'ajout du
bruit
% pour l'ajout du bruit on cherche les indices correspondants
aux
% differents types de bruit
indicebruit=cell(1,nbtraitement);
for ib=1:(nbtraitement-1)
    indicebruit{ib}=find(typebruitetat==ib);
end
indicebruit{nbtraitement}=find(typebruitetat>=nbtraitement);

% Initialisation des particules %%%%%%%%%%%%
% en utilisant les lois a priori
[Xt,Xmin,Xmax]=pi0(n);
% Tableau de resultats
Moyenne=zeros(size(Xt,1),nbobs);
ET=Moyenne;
ICInf=Moyenne;
ICSup=Moyenne;

%% Boucle sur le temps
for ti=SupportT(1):deltaT:SupportT(2)
    % Evolution des particules
    Xt=fetat(Xt,ti,1);%typebruitetat);
    %% Si on se trouve a un point d'observation
    if(any(abs(tobs-ti)<epsilon))
        % on cherche le numero (indice) de l'observation
        indiceobs=find(abs(tobs-ti)<epsilon);
        disp(['indice observation en cours = '
num2str(indiceobs)])
        % calcul de l'observation theorique
        Ytheorique=hobsbruit(Xt,paramobs(:,:,indiceobs)); %
(qxn)
        %% Calcul des poids
        % differences entre les estimes (Ytheorique) et les
observations (qxn)
        diff=(Ytheorique-repmat(yobs(:,indiceobs),1,n));
        clear Ytheorique
        % ecart-type de ces differences (qx1)
        covdiff=std(diff,0,2);
        % Largeur de la fenetre du noyau (qx1)
        hn=covdiff./(n^(1/(4+length(covdiff))));
        clear covdiff
        % on l'utilise pour normaliser diff
        diffnormalise=diag(1./hn)*diff;
        clear diff hn
        % Poids
        poids=exp(-0.5*sum(diffnormalise.^2,1));%./prod(hn)
```

```
        clear diffnormalise
        % ajustement pb numerique
        if(all(poids==0))
            poids=ones(size(poids));
            %disp('pb avec les poids')
        end
        % normalisation des poids
        poids=poids/sum(poids);
        %% Echantillonnage sur les particules de poids positif
        ind=find(poids>0);
        %disp([num2str(length(ind)) ' particules gardees'])
        %% Stockage des resultats au temps
        Moyenne(:,indiceobs)=Xt*(poids(:));
        ET(:,indiceobs)=sqrt(var(Xt,poids,2));%
        % Calcul des IC a 95%
        [TAB,Indice]=sort(Xt,2);
        for j=1:size(Xt,1)
            w=poids(Indice(j,:));
            w=cumsum(w);
            II=find(w>=0.025 | w<= 0.975);%find(w>=0.005 | w<=
0.995);%
            if isempty(II)
                II=[1 n];
            end
            ICInf(j,indiceobs)=TAB(j,II(1));
            ICSup(j,indiceobs)=TAB(j,II(end));
        end
        clear TAB Indice w II
        %% Tirage multinomial
        ind=randsample(ind,n,true,poids(ind));
        Xt=Xt(:,ind);
        clear poids ind
        Xtempo=Xt; % sauvegarde des particules
        %% Rajout d'un bruit apres re-echantillonage (etape de
regularisation)
        % Calcul de l'ecart type  de la perturbation
        ETPerturbation=std(Xtempo,0,2);
        ETPerturbation=ETPerturbation/(n^(1/5));
        % pour toutes les particules
        particulesachanger=1:n;
        % tant qu'elles ne sont pas dans [Xmin,Xmax]
        while(~isempty(particulesachanger))
            % variance dependante du nombre de particule et de
l'etat
            % Diversification des particules par
regularisation
            %case 1 % binomial
            if (~isempty(indicebruit{1}))

bruit=round(diag(ETPerturbation(indicebruit{1}))*randn(length(
indicebruit{1}),length(particulesachanger)));
```

```
Xt(indicebruit{1},particulesachanger)=Xtempo(indicebruit{1},pa
rticulesachanger)+bruit;
                clear bruit
            end
            %case 2 % poisson
            if (~isempty(indicebruit{2}))
                ETPert=ETPerturbation(indicebruit{2}).^2;

ETPert=repmat(ETPert,1,length(particulesachanger));
                bruit=poissrnd(ETPert)-round(ETPert);

Xt(indicebruit{2},particulesachanger)=Xtempo(indicebruit{2},pa
rticulesachanger)+bruit;
                clear ETPert bruit
            end
            %case 3 % gaussien
            if (~isempty(indicebruit{3}))
                bruit =
diag(ETPerturbation(indicebruit{3}))*randn(length(indicebruit{
3}),length(particulesachanger));

Xt(indicebruit{3},particulesachanger)=Xtempo(indicebruit{3},pa
rticulesachanger)+bruit;
                clear bruit
            end
            % on cherche les particules qui ne sont pas dans
le domaine de recherche

particulesachanger=find(any(Xt<Xmin,1)+any(Xmax<Xt,1));
        end % while(~isempty(particulesachanger))
        clear particulesachanger Xtempo
        if (affichage)
            for numparam=1:4
                figure(numparam)
                hist(Xt(numparam,:),50)
            end
        end
    end % if(any(abs(tobs-ti)<epsilon))
end % for ti=SupportT(1):deltaT:SupportT(2)

% on efface les variables temporaires pour gain temps de
calcul
clear Xt Xmin Xmax
%ETPerturbation Xtempo Ytheorique diff diffnormalise TAB
Indice w poids particulesachanger
```

Exemple de données : fichier EXCEL

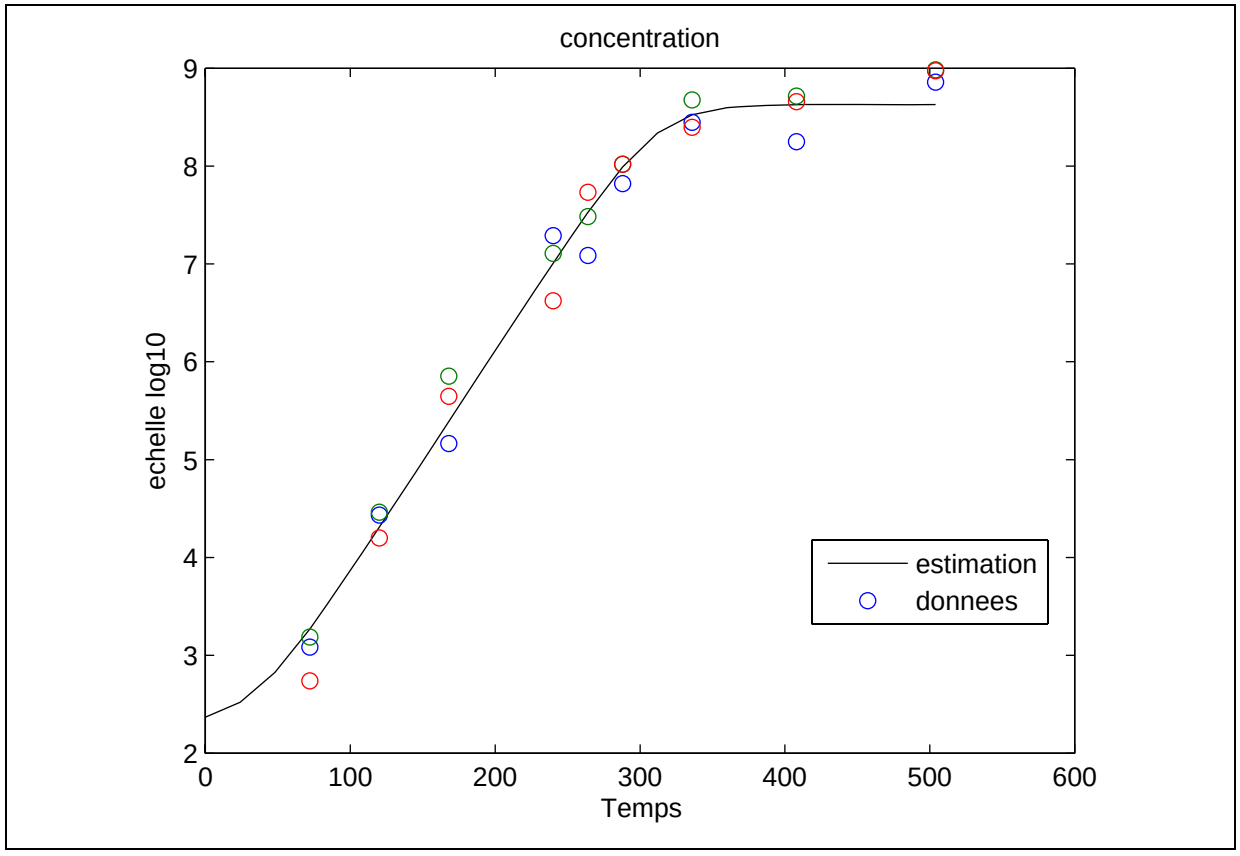
t (h)	m (g) (masse analysée)	a (facteur de dilution initial - suspension mère)	n (nb dilutions en tubes)	F (facteur de dilution en tubes total)	v (ml) (volume de la dernière dilution ensemencé)	Nb colonies observées	concentration (ufc/g)	log concentration
0	10	0,1	0	1	0,6	8	133,3	2,12
0	10	0,1	0	1	0,6	19	316,7	2,50
0	10	0,1	0	1	0,6	13	216,7	2,34
72	10	0,1	0	1	0,6	73	1216,7	3,09
72	10	0,1	0	1	0,6	92	1533,3	3,19
72	10	0,1	0	1	0,6	33	550,0	2,74
120	10	0,1	0	1	0,05	136	27200,0	4,43
120	10	0,1	0	1	0,05	145	29000,0	4,46
120	10	0,1	0	1	0,05	79	15800,0	4,20
168	10	0,1	1	0,1	0,05	73	146000,0	5,16
168	10	0,1	1	0,1	0,05	357	714000,0	5,85
168	10	0,1	1	0,1	0,05	222	444000,0	5,65
240	10	0,1	2	1,00E-03	0,05	97	19400000,0	7,29
240	10	0,1	2	1,00E-03	0,05	64	12800000,0	7,11
240	10	0,1	2	1,00E-03	0,05	21	4200000,0	6,62
264	10	0,1	2	1,00E-03	0,05	61	12200000,0	7,09
264	10	0,1	2	1,00E-03	0,05	152	30400000,0	7,48
264	10	0,1	2	1,00E-04	0,05	27	54000000,0	7,73
288	10	0,1	2	1,00E-04	0,05	33	66000000,0	7,82
288	10	0,1	2	1,00E-04	0,05	52	104000000,0	8,02
288	10	0,1	2	1,00E-04	0,05	52	104000000,0	8,02
336	10	0,1	2	1,00E-04	0,05	140	280000000,0	8,45
336	10	0,1	2	1,00E-04	0,05	237	474000000,0	8,68
336	10	0,1	2	1,00E-04	0,05	125	250000000,0	8,40
408	10	0,1	2	1,00E-04	0,05	89	178000000,0	8,25
408	10	0,1	3	1,00E-05	0,05	26	520000000,0	8,72
408	10	0,1	2	1,00E-04	0,05	228	456000000,0	8,66
504	10	0,1	3	1,00E-05	0,05	36	720000000,0	8,86
504	10	0,1	3	1,00E-05	0,05	48	960000000,0	8,98
504	10	0,1	3	1,00E-05	0,05	47	940000000,0	8,97

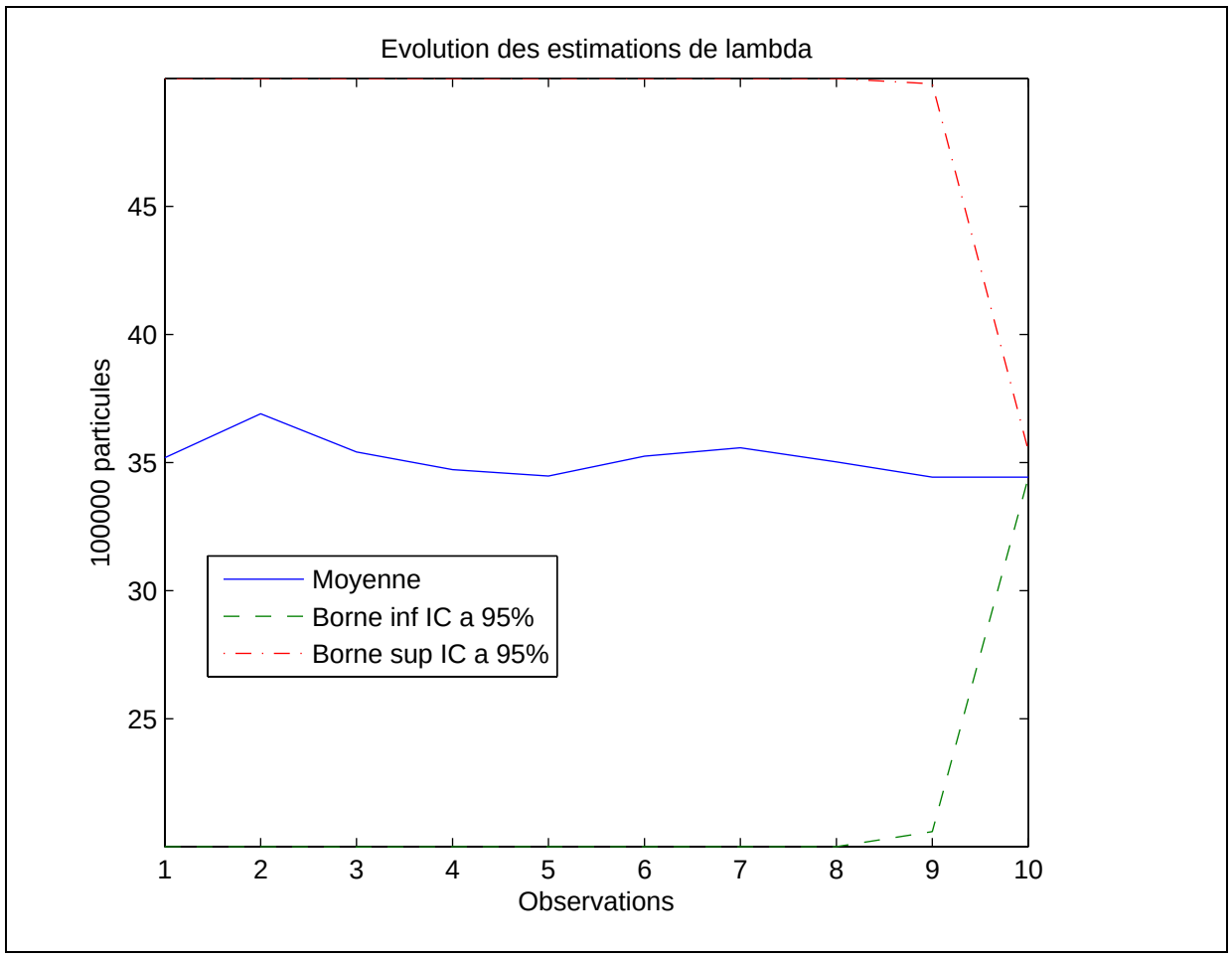
Listing de sortie pour le modèle de BARANYI

```
>> mainmanipJCA


On dispose de 10 temps de mesures et de 3 repetitions
de chaque mesure.
indice observation en cours = 1
indice observation en cours = 2
indice observation en cours = 3
indice observation en cours = 4
indice observation en cours = 5
indice observation en cours = 6
indice observation en cours = 7
indice observation en cours = 8
indice observation en cours = 9
indice observation en cours = 10

Elapsed time is 65.733841 seconds.

Estimations obtenues pour les parametres,
dans l'ordre [mumax;lambda;N0;Nmax] :

mumax  = 0.052
lambda = 34.43
N0     = 234.47
Nmax   = 425480063
```

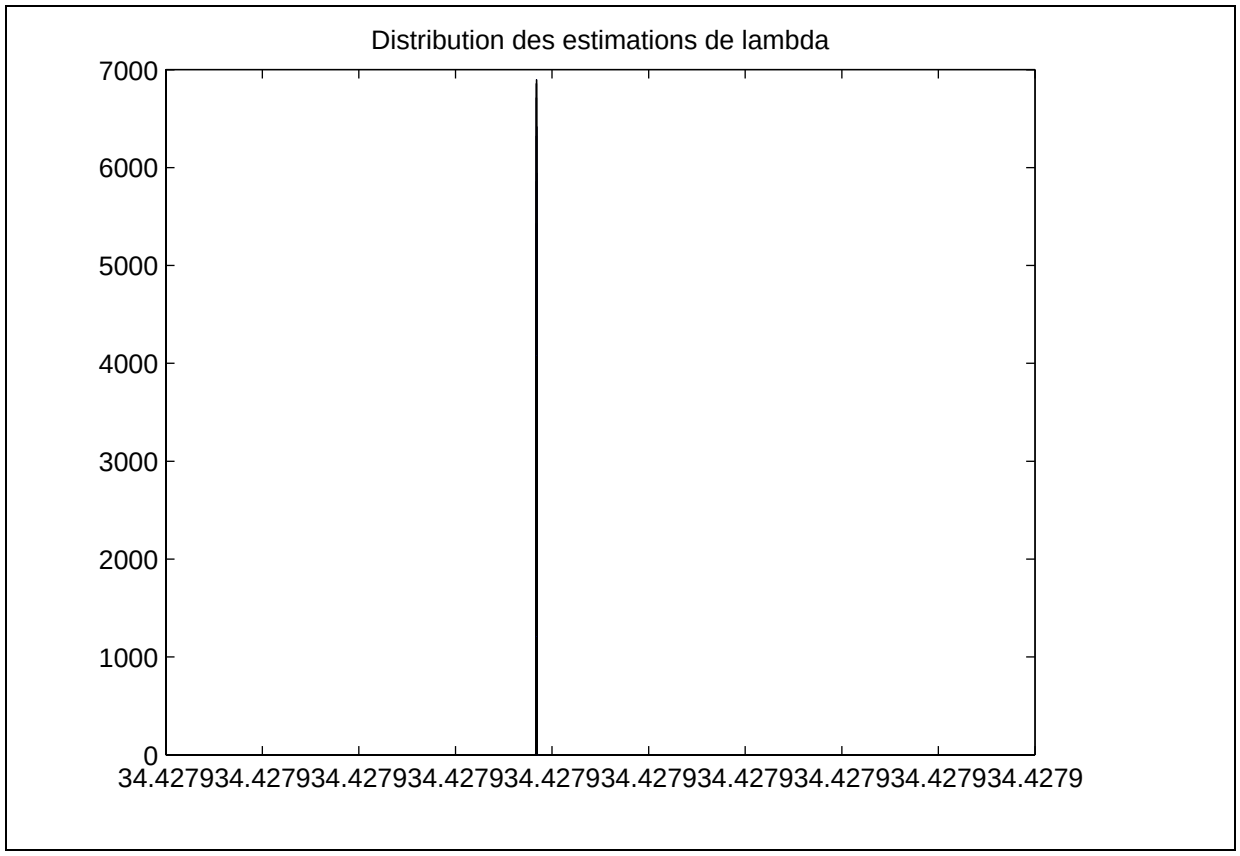
Graphiques pour le modèle de BARANYI

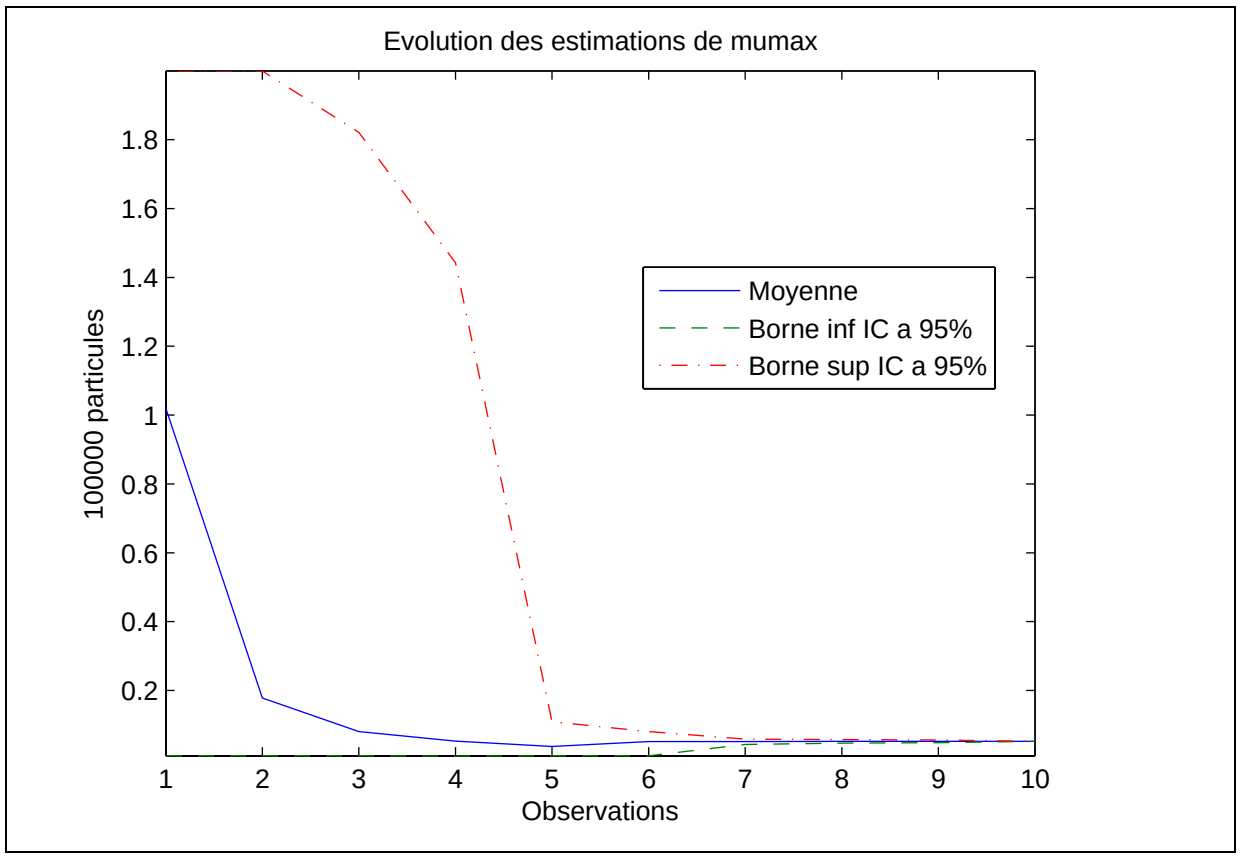
Graphique de la dynamique à partir des valeurs des estimations des paramètres obtenues à la fin des itérations



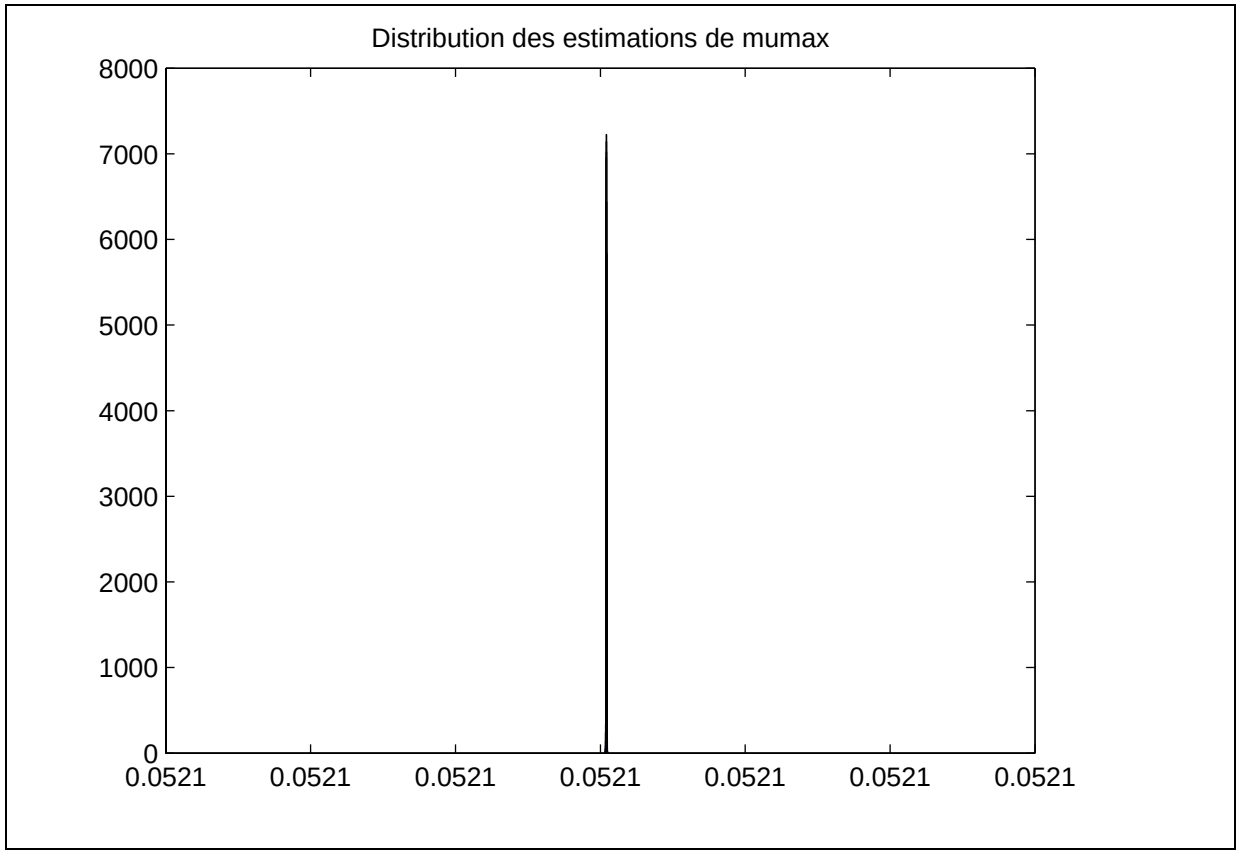

Evolution des estimations de lambda
100000 particules
45
40
35
30
25
1
2
3
4
5
6
7
8
9
10
Observations
Moyenne
Borne inf IC a 95%
Borne sup IC a 95%

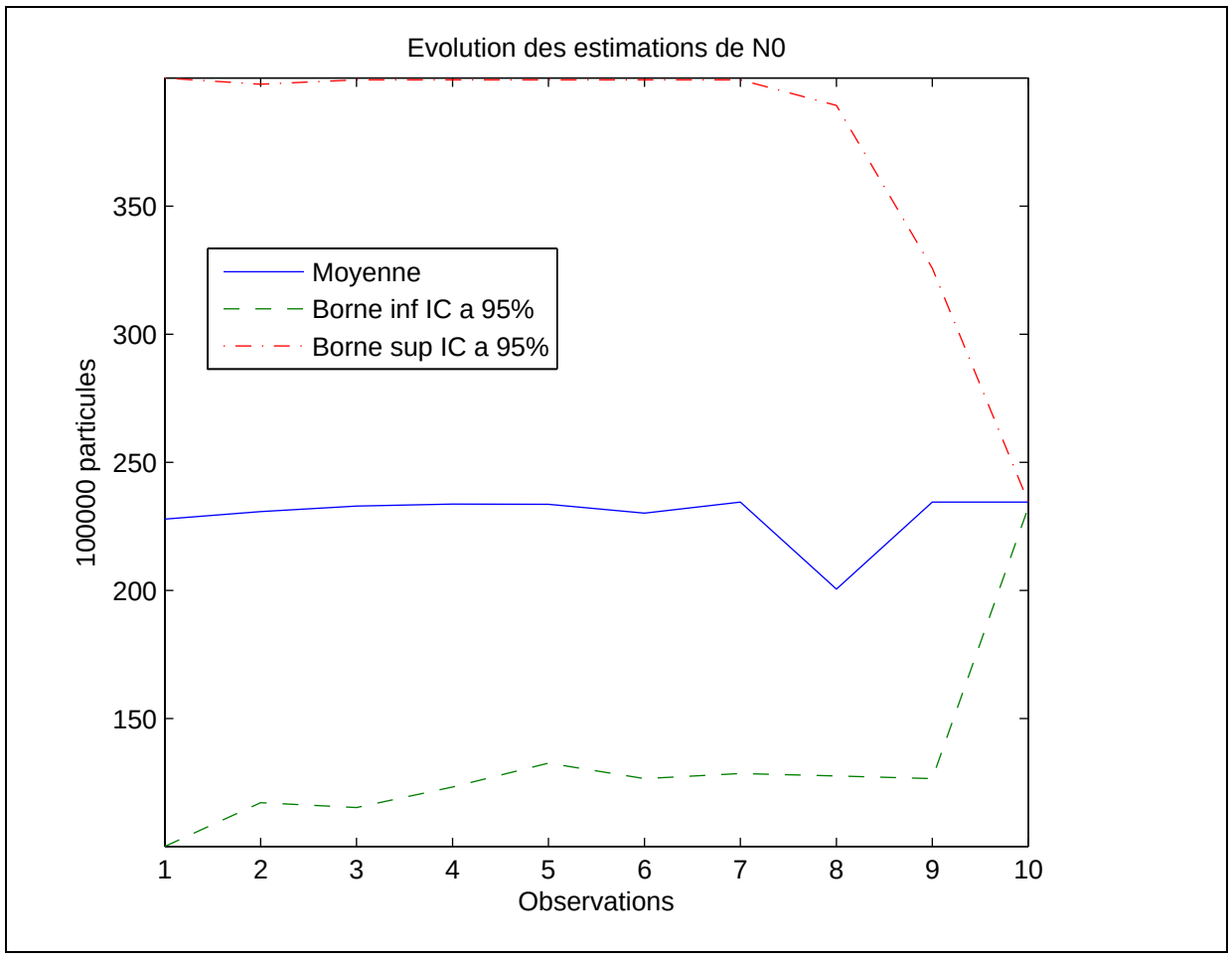
Histogramme des estimations
du paramètre lambda à la fin des itérations



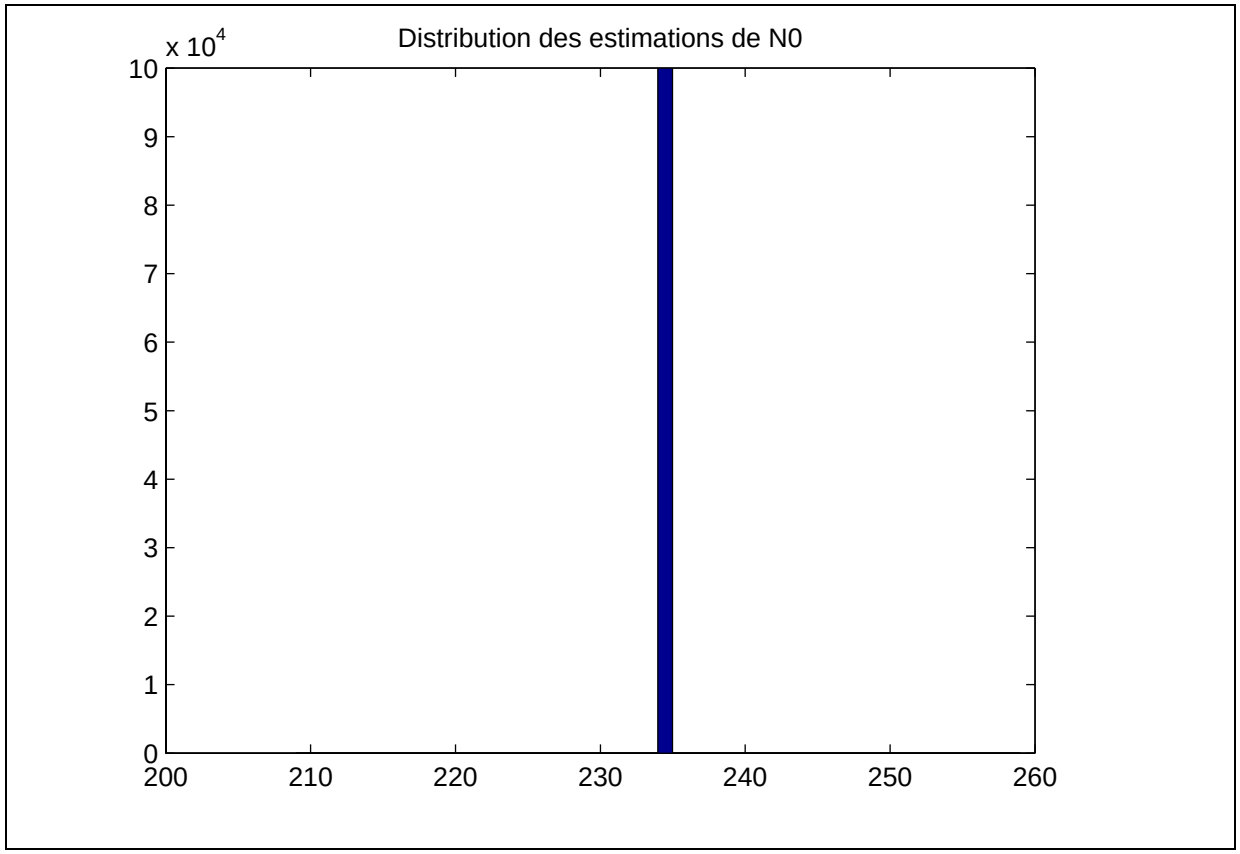

Evolution des estimations de mumax
100000 particules
Observations
Moyenne
Borne inf IC a 95%
Borne sup IC a 95%
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
1
2
3
4
5
6
7
8
9
10

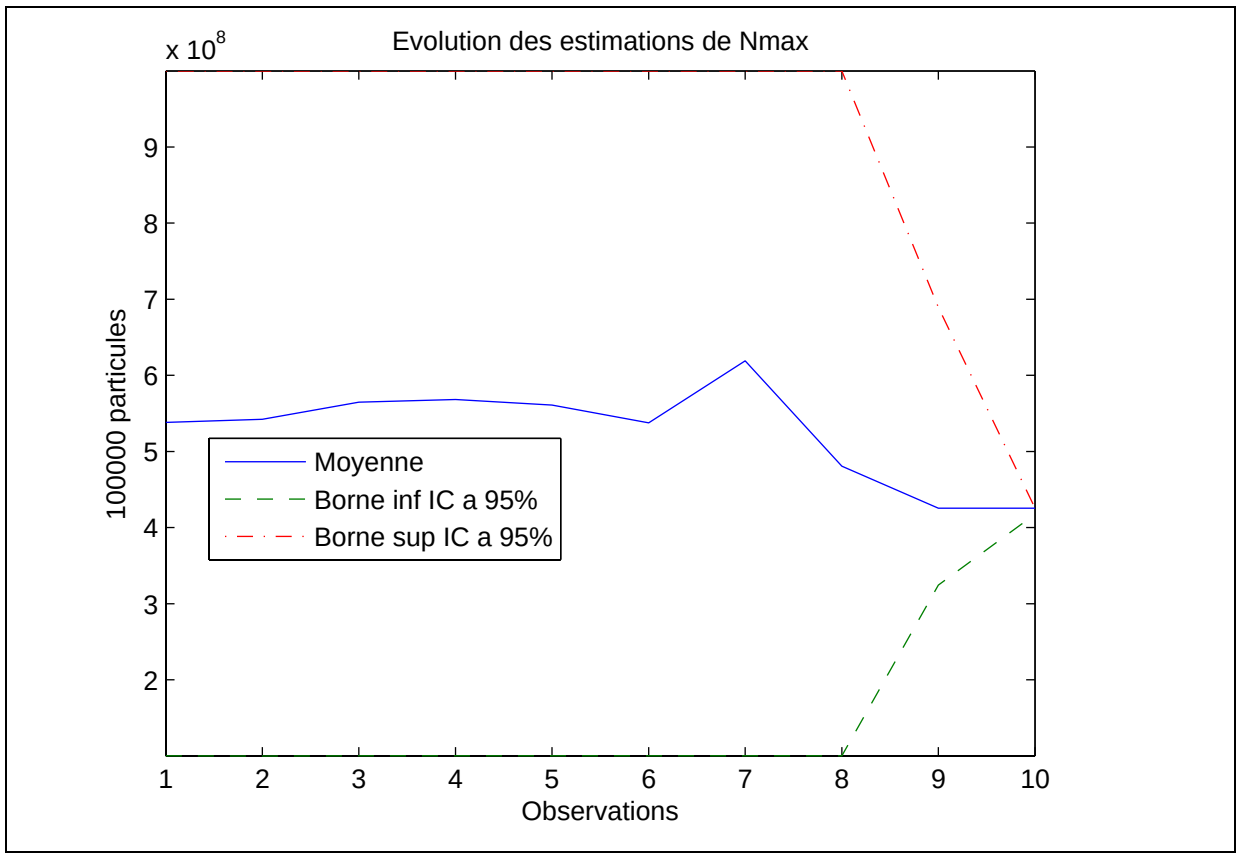
Histogramme des estimations
du paramètre mumax à la fin des itérations



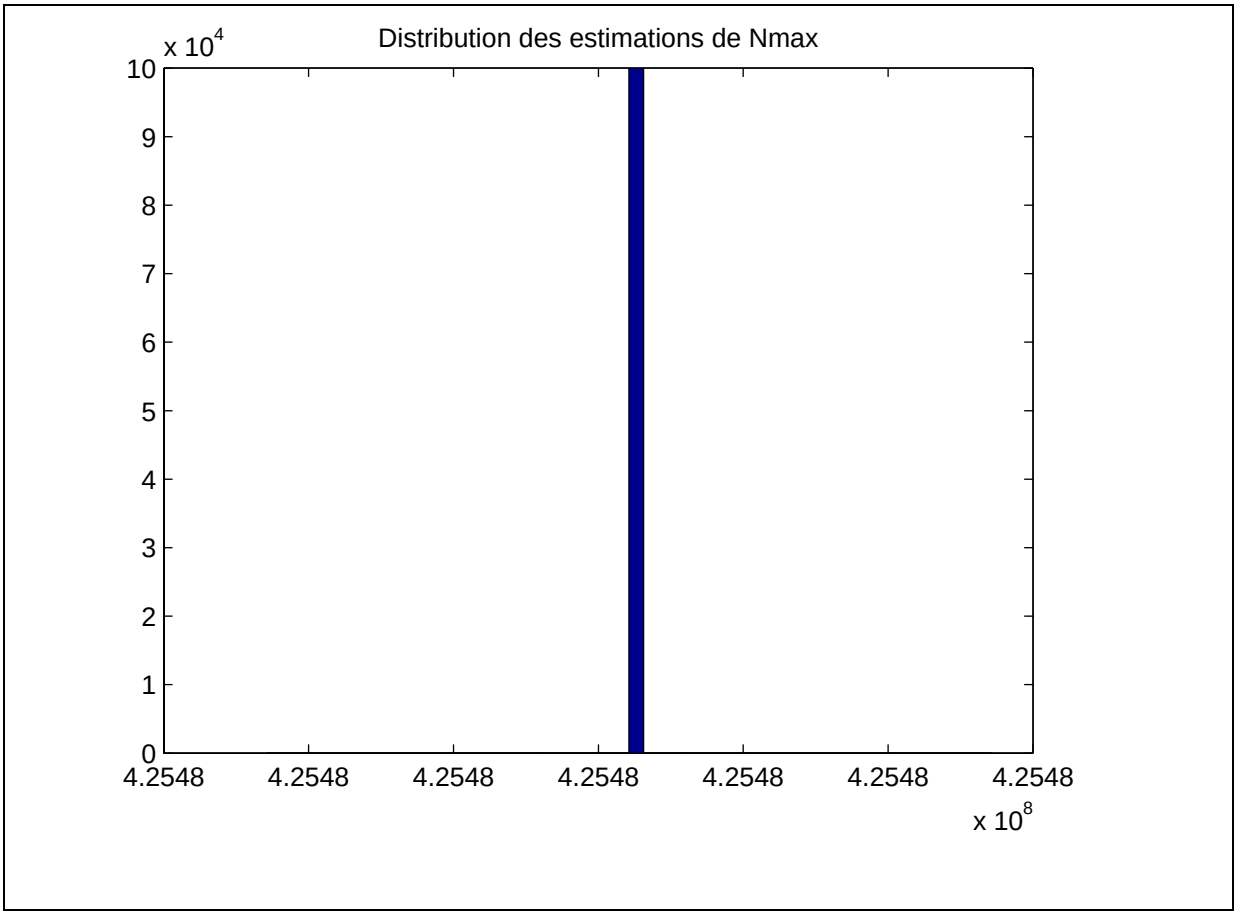

Evolution des estimations de N0
100000 particules
350
300
250
200
150
Moyenne
Borne inf IC a 95%
Borne sup IC a 95%
1
2
3
4
5
6
7
8
9
10
Observations

Histogramme des estimations
du paramètre N0 à la fin des itérations




Evolution des estimations de Nmax
x 10^8
100000 particules
Observations
Moyenne
Borne inf IC a 95%
Borne sup IC a 95%
1
2
3
4
5
6
7
8
9
10
2
3
4
5
6
7
8
9

Histogramme des estimations
du paramètre Nmax à la fin des itérations

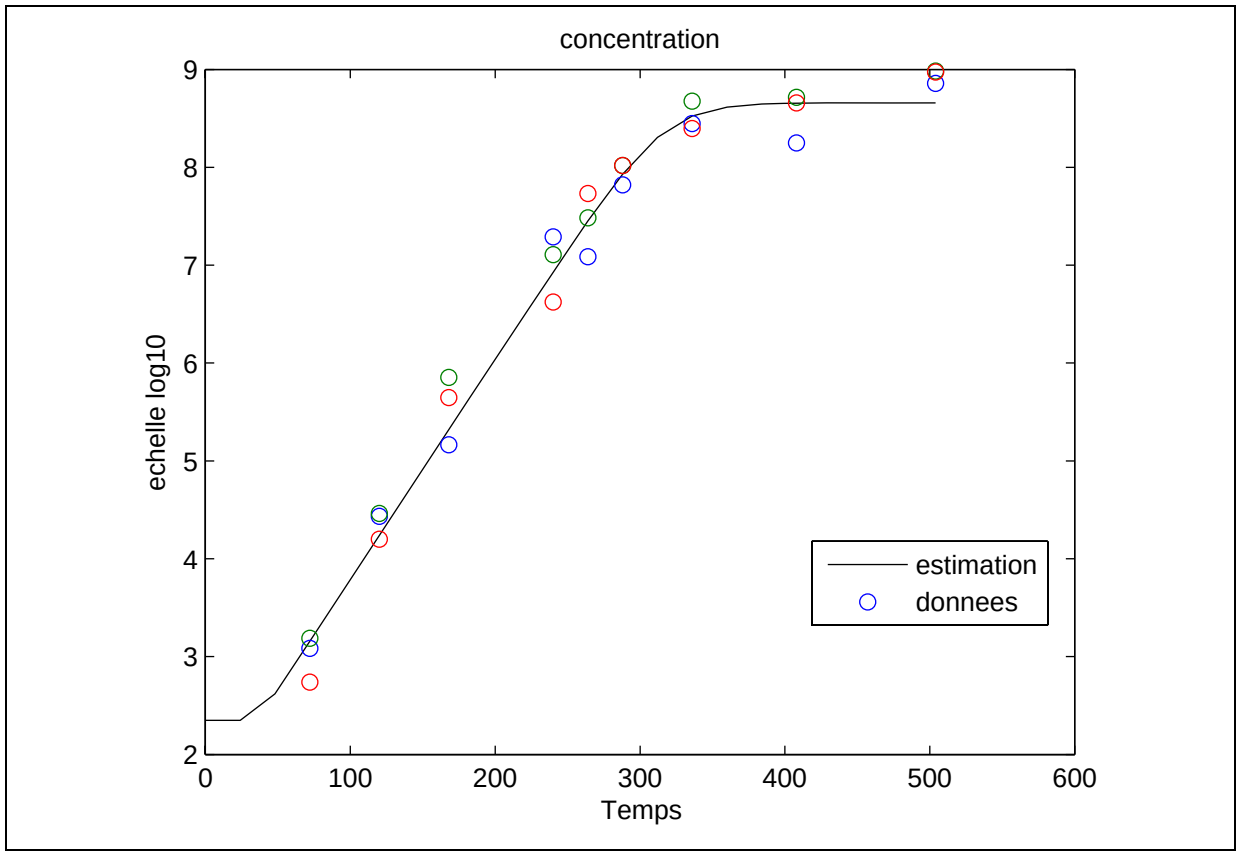


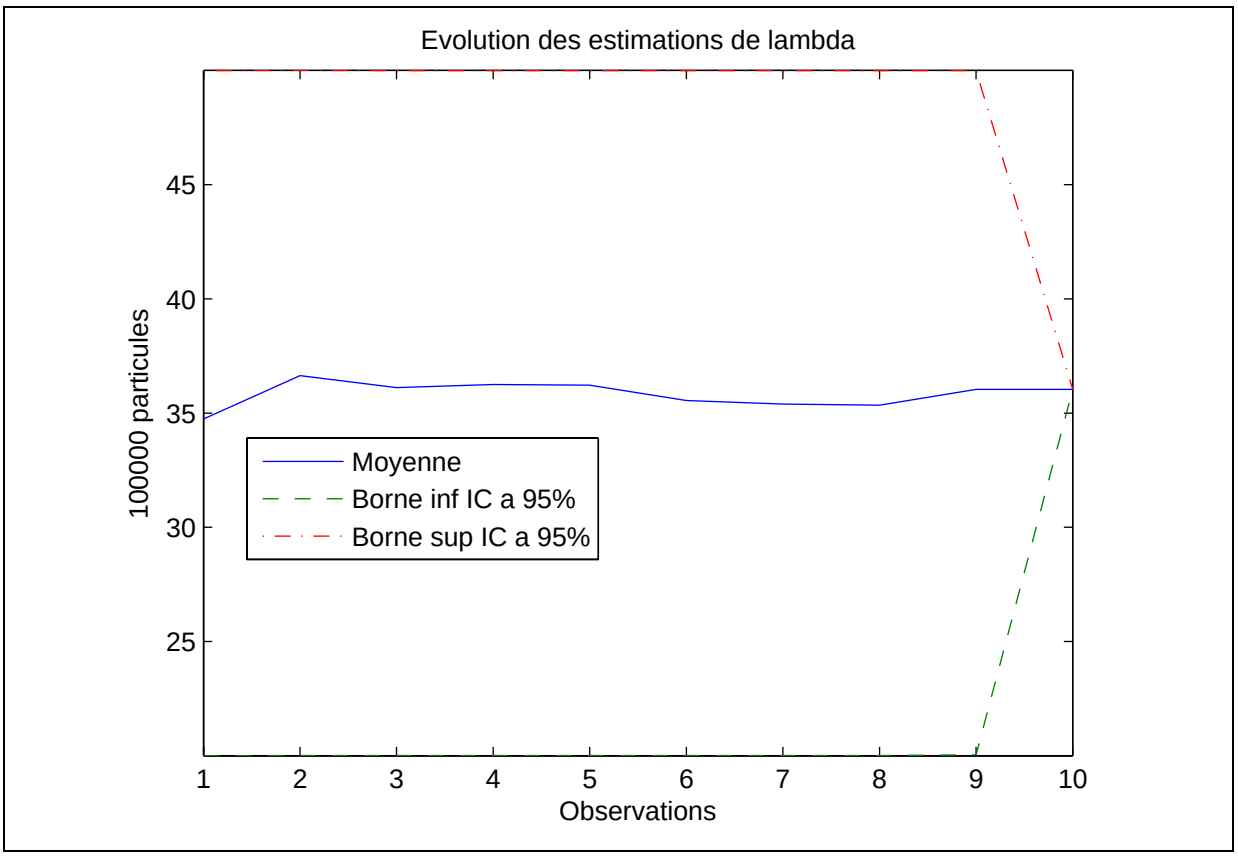
Listing de sortie pour le modèle de ROSSO

```
>> mainmanipJCA


On dispose de 10 temps de mesures et de 3 repetitions
de chaque mesure.
indice observation en cours = 1
indice observation en cours = 2
indice observation en cours = 3
indice observation en cours = 4
indice observation en cours = 5
indice observation en cours = 6
indice observation en cours = 7
indice observation en cours = 8
indice observation en cours = 9
indice observation en cours = 10

Elapsed time is 64.337832 seconds.

Estimations obtenues pour les parametres,
dans l'ordre [mumax;lambda;N0;Nmax] :

mumax=0.0522
lambda=36.036
N0=223.21
Nmax=456090587
```

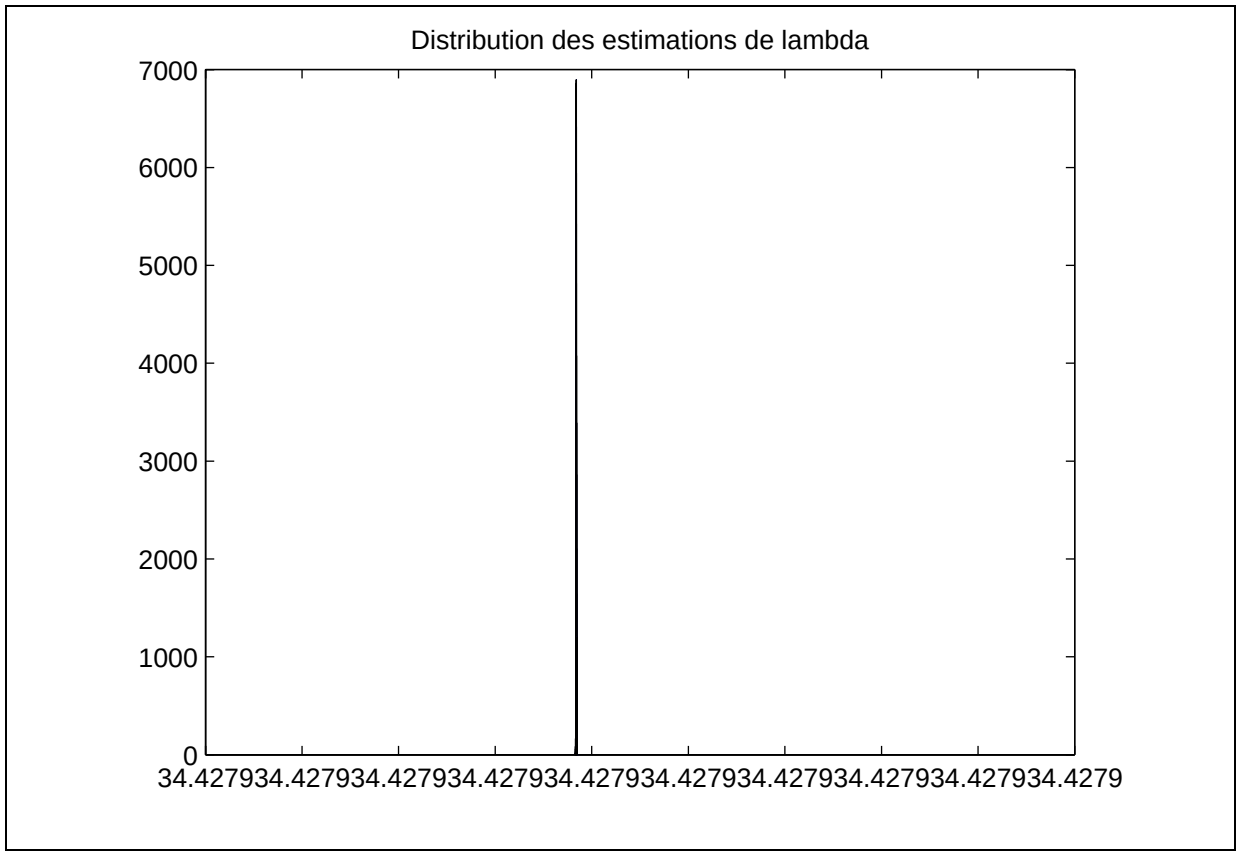
Graphiques pour le modèle de ROSSO

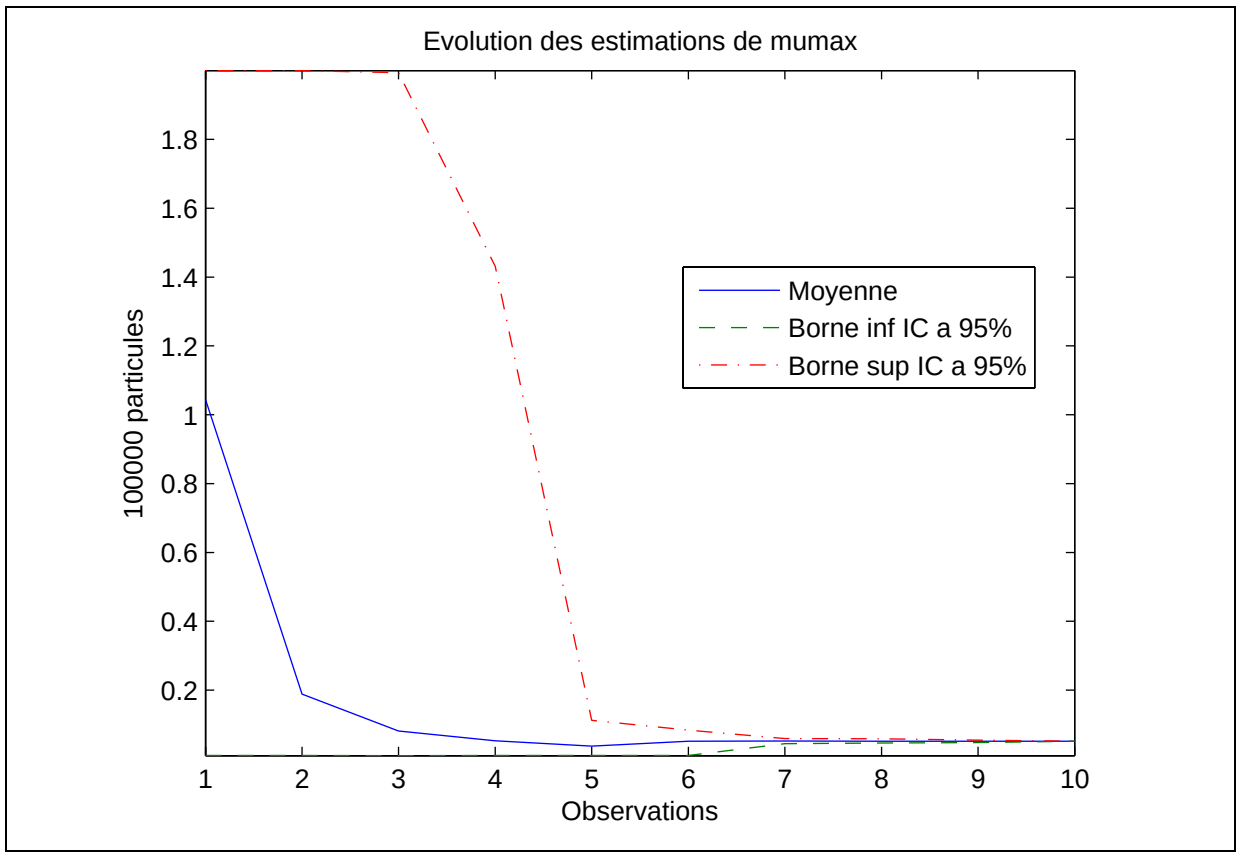
Graphique de la dynamique à partir des valeurs des estimations des paramètres obtenues à la fin des itérations



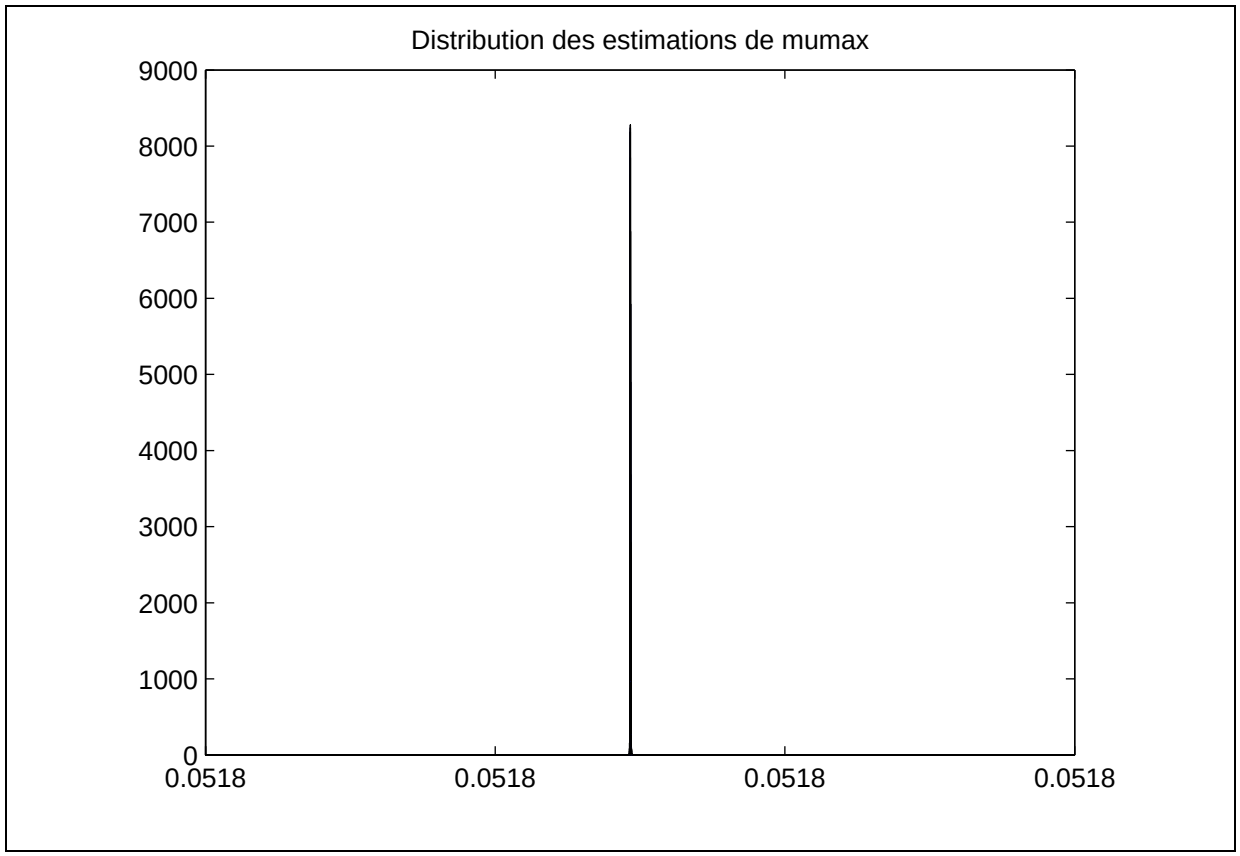

Evolution des estimations de lambda
100000 particules
45
40
35
30
25
1
2
3
4
5
6
7
8
9
10
Observations
Moyenne
Borne inf IC a 95%
Borne sup IC a 95%

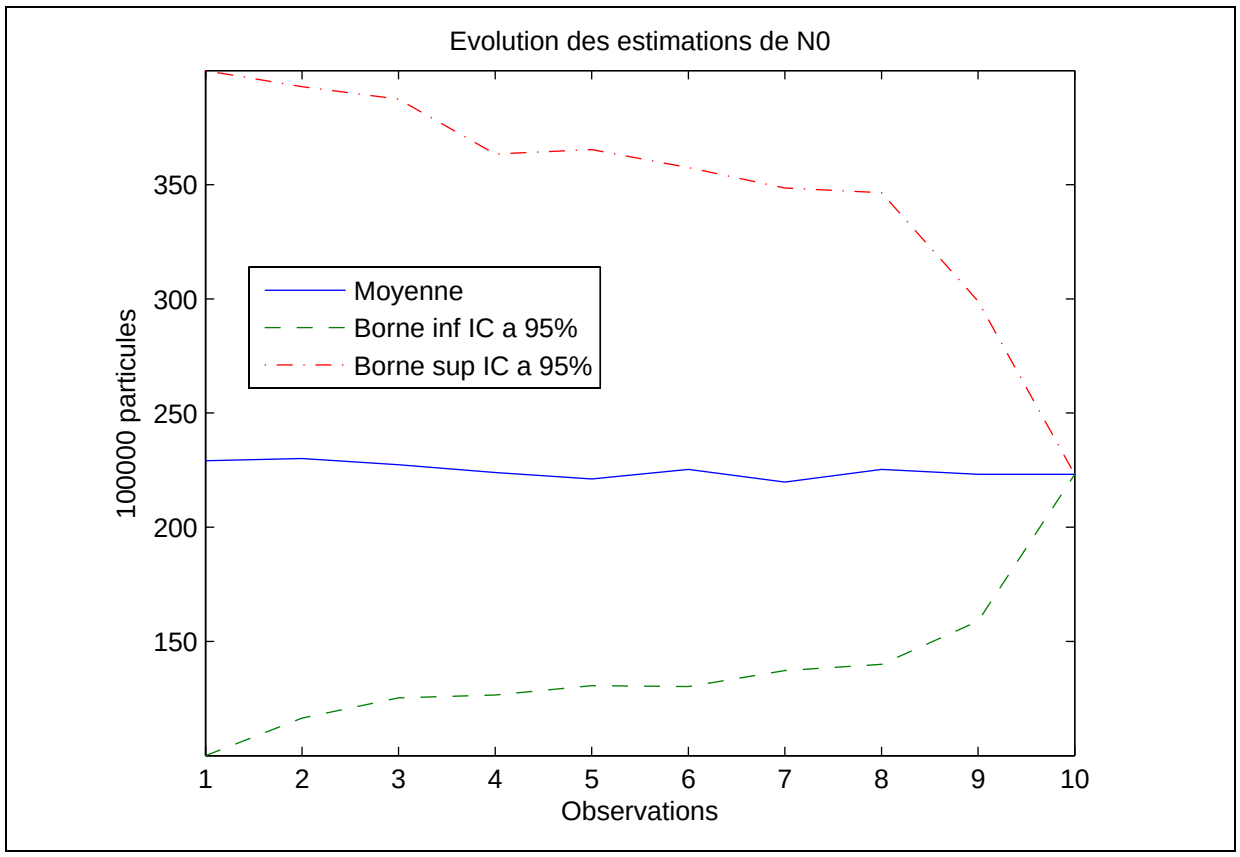
Histogramme des estimations
du paramètre lambda à la fin des itérations



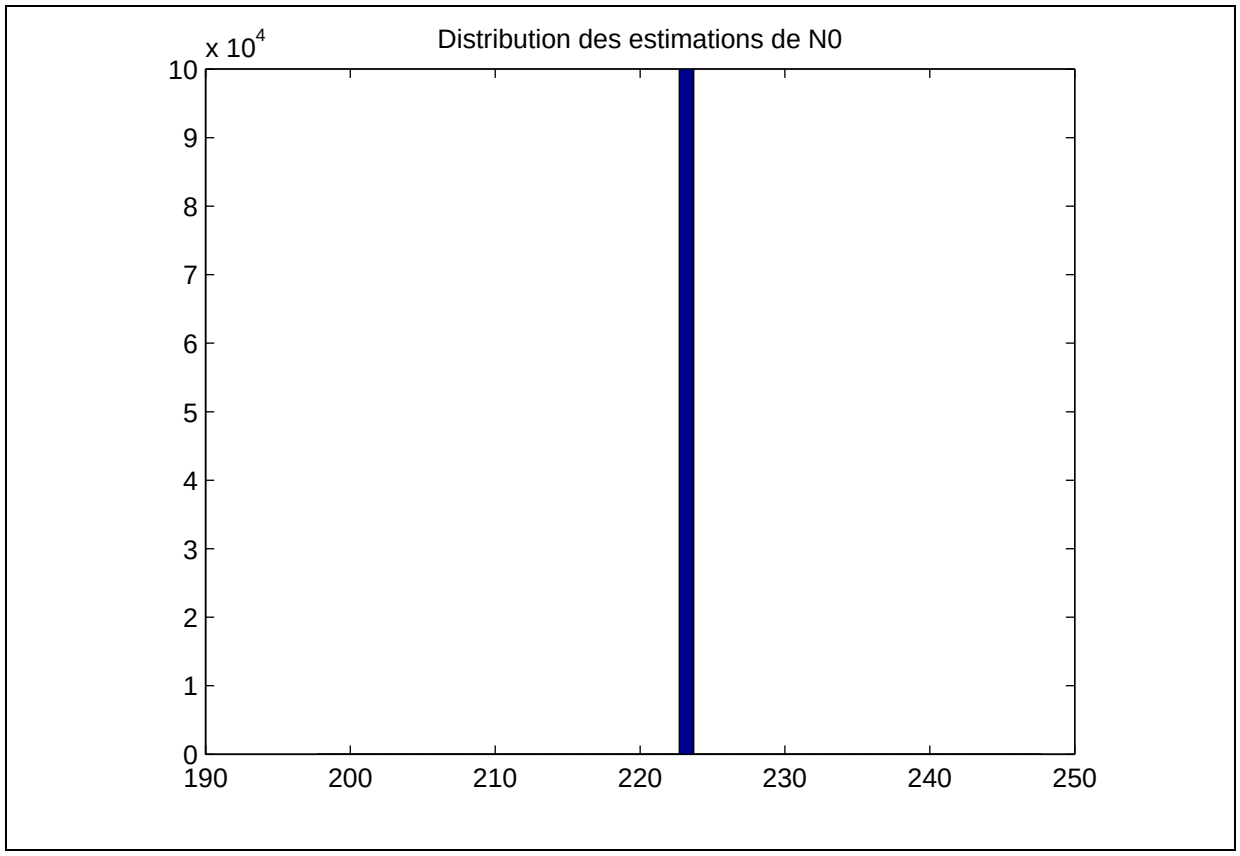

Evolution des estimations de mumax
100000 particules
Observations
Moyenne
Borne inf IC a 95%
Borne sup IC a 95%
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
1
2
3
4
5
6
7
8
9
10

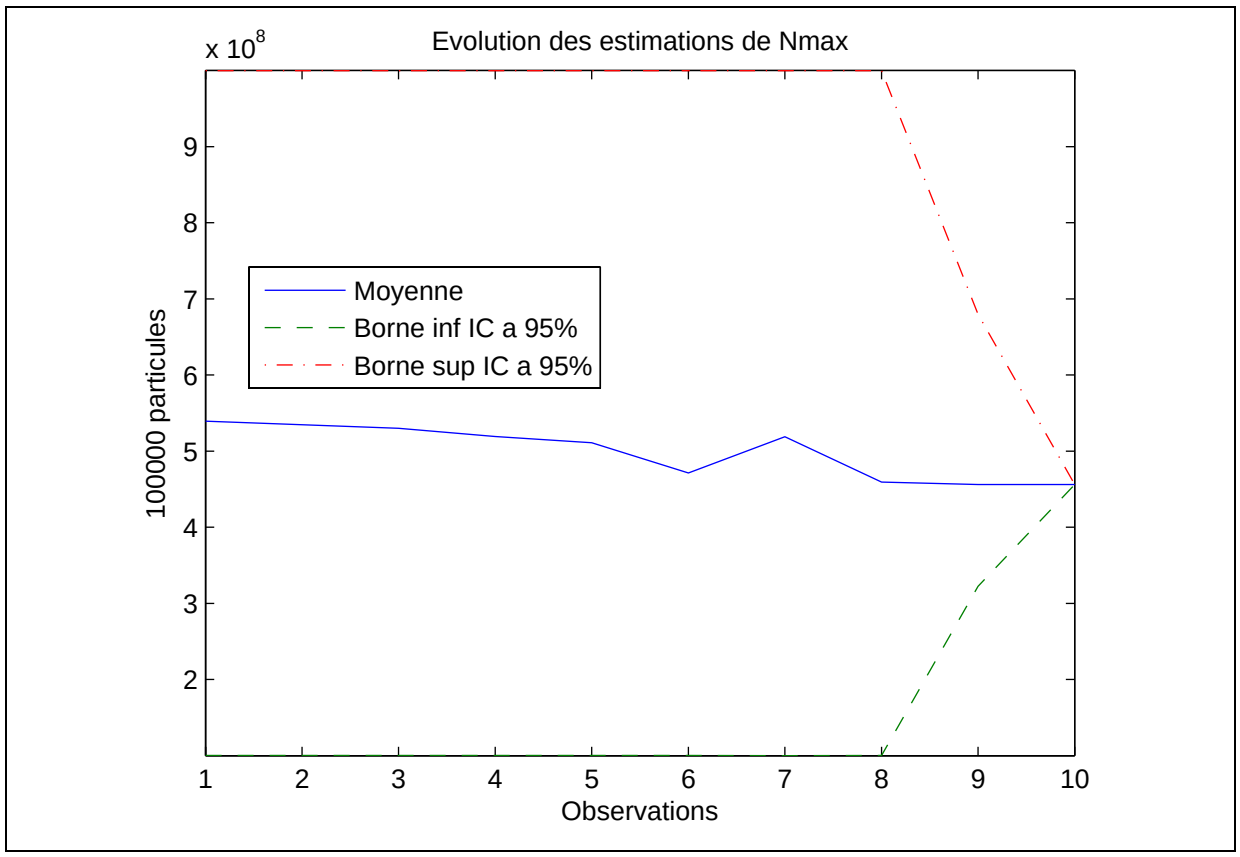
Histogramme des estimations
du paramètre mumax à la fin des itérations




Evolution des estimations de N0
100000 particules
350
300
250
200
150
1
2
3
4
5
6
7
8
9
10
Observations
Moyenne
Borne inf IC a 95%
Borne sup IC a 95%

Histogramme des estimations
du paramètre N0 à la fin des itérations




Evolution des estimations de Nmax
x 10^8
100000 particules
Observations
Moyenne
Borne inf IC a 95%
Borne sup IC a 95%
1
2
3
4
5
6
7
8
9
10

Histogramme des estimations
du paramètre Nmax à la fin des itérations