Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Lecture with Computer Exercises: Modelling and Simulating Social Systems with MATLAB

Project Report

Division of labour in insect societies

Anna Stamp & Kan Wang

Zürich
December 2010

Eigenständigkeitserklärung

Hiermit erkläre ich, dass ich diese Gruppenarbeit selbständig verfasst habe, keine anderen als die angegebenen Quellen-Hilfsmittel verwenden habe, und alle Stellen, die wörtlich oder sinngemäß aus veröffentlichen Schriften entnommen wurden, als solche kenntlich gemacht habe. Darüber hinaus erkläre ich, dass diese Gruppenarbeit nicht, auch nicht auszugsweise, bereits für andere Prüfung ausgefertigt wurde.

Anna Stamp Kan Wang

Agreement for free-download

We hereby agree to make our source code of this project freely available for download from the web pages of the SOMS chair. Furthermore, we assure that all source code is written by ourselves and is not violating any copyright restrictions.

Anna Stamp　　　　Kan Wang

Table of content

Individual contributions

- Development of model understanding
  - FTM model in Bonabeau et al. (1996): Anna
  - VTM model in Theraulaz et al. (1998): Kan
- Implementation in Matlab
  - FTM model in Bonabeau et al. (1996): Anna (& Kan)
  - VTM model in Theraulaz et al. (1998): Kan
- Development of the adaptation structure: Anna & Kan
- Implementation of adaptation in structure: Kan
- Report writing: Anna & Kan

Introduction and Motivations

Many species of social insects show a division of labour in colonial life. The workers show "behavioural flexibility, that is, they can "switch tasks in response to internal perturbations or external challenges" (Bonabeau et al. 1996), which are related to a stimulus or set of stimuli. An example for such a "task" workers of a colony have to fulfil is larval feeding, which has the associated stimulus larval demand (expressed e.g. through emissions of pheromones).
The question how insects decide if they switch task or engage in task performance is addressed in the two reviewed paper. The first paper, (Bonabeau et al. 1996), elaborates on a simple model, which was introduced by several authors (Robinson 1987a; Robinson 1987b; Calabi 1988 ; Robinson 1992). It describes the division of labour based on fixed response thresholds of individuals to the stimuli and can explain how the workers' behavioural flexibility can account for the flexibility at the colony level and the "model can constitute the basic mechanism for a theory of the regulation of division of labour in insect societies" (FTM: *Fixed Threshold Model*) (Bonabeau et al. 1996, p.1566). The second paper, (Theraulaz et al. 1998), introduces learning and forgetting in the model based on a reinforcement process, which allows thresholds to vary in time (VTM: *Variable Threshold Model*). Thus, the VTM can account for the genesis of task allocation (that is: changing roles of individuals) and "robust task specialization within castes", while the FTM assumes that individuals are differentiated and roles predetermined.

Simulating the division of labour in insect societies can help to explain and thus better understand experimental observations. This report shall document how both models can be implemented in MATLAB and further focus on effects of parameter choices in the VTM. Different learning and forgetting rates are chosen to specify the turning point of the division and specification of labour. A minimal learning factor is implemented in the model to guarantee the division of labour.

Description of the Models

FTM

When the stimulus intensity exceeds the worker's individual threshold concerning this task, the worker will start performing the task. Since thresholds are fixed and no learning is assumed, these models can only be used for approximation of short time-scales, when thresholds can be considered constant. In the paper of (Bonabeau et al. 1996), some experimental observations with Pheidole (a genus of ants), are quantitatively simulated with the FTM.
Two cases are discussed in the paper, first FTM with one task and two distinct castes and second FTM with two tasks and two distinct castes. We limit our explanations here to the first case, since it is sufficient to clarify the basic principles of the FTM. The model can be denoted with formula (1) to (3) listed below.

Formula (1) explains the probability that an individual of caste *i* starts performing the task (*X* refers to the state of the individual, with 0 corresponding to inactivity and 1 to performing the task). θ_i is the response threshold of caste *i* and *s* corresponds to the "magnitude of the task, that affects the probability of being exposed to it".

$$P_i(X = 0 \rightarrow X = 1) = \frac{s^2}{s^2 + \theta_i^2} \tag{1}$$

Formula (2) describes *p*, the probability that an active individual of caste *i* stops performing a task and becomes inactive. *p* can be found experimentally, since *1/p* is the "average time spent by an individual in task performance before giving up this task" (Bonabeau et al. 1996, p. 1567).

$$P_i(X = 1 \rightarrow X = 0) = p \tag{2}$$

The last equation of the FTM (3) is describing the evolution of stimulus intensity *s* (in discrete time). N_i is the number of active individuals belonging to caste *i*, *N* is the total number of potentially active individuals in the colony, δ is the increase in stimulus intensity per unit time, and α is a scale factor measuring the efficiency of task performance, which is assumed to be identical for individuals of all castes.

$$s(t+1) = s(t) + \delta - \frac{\alpha}{N}(N_1 + N_2) \tag{3}$$

VTM

In the following section the set of equations constituting the VTM is introduced. For a more detailed explanation of the derivation of these formulas please refer to the original document.
The first equation of the VTM (4) is similar to formula (1) of the FTM, however, the

denotation is slightly different in order to account for different tasks and associated stimuli (j=1,..., m).

$$T_{\theta_{ij}} = \frac{s_j^2}{s_j^2 + \theta_{ij}^2} \tag{4}$$

The threshold θ_{ij}, referring to individual i and task j, is updated in a self-reinforcing way. "The more individual i performs task j, the lower θ_{ij}, and vice versa" (Theraulaz et al. 1998).These processes are denoted by the coefficients $\xi(\Delta t)$ (learning) and $\varphi(\Delta t)$ (forgetting), while Δt refers to the time performing (learning) respectively not performing (forgetting) a task. ξ and φ are assumed to be identical for all tasks.
The fraction of time spent by individual i in task j performance is described by x_{ij} and the dynamics of θ_{ij} is restricted to an interval [θ_{min}, θ_{max}]. The average temporal dynamics of θ_{ij} is then given by (5), of x_{ij} by (6), and of s_{ij} by (7):

$$d_i\theta_{ij} = \left[\left(1 - x_{ij}\right)\varphi - x_{ij}\xi\right]\Theta\left(\theta_{ij} - \theta_{min}\right)\Theta\left(\theta_{max} - \theta_{ij}\right) \tag{5}$$

$$d_i x_{ij} = T_{\theta_{ij}}\left(s_j\right)\left(1 - \sum_{k=1}^{m} x_{ik}\right) - p x_{ij} + \psi(i,j,k) \tag{6}$$

$$\mathrm{d_i s_{ij}} = \delta - \frac{\alpha}{\mathrm{N}}\left(\sum_{\mathrm{i}=1}^{\mathrm{N}} \mathrm{x_{ij}}\right) \tag{7}$$

The last term of formula (5) is a step function, which is used in order to maintain θ within model boundaries ($\Theta(y) = 0$ if $y \leq 0, \Theta(y) = 1$ if $y > 0$). The last term in formula (6) is a "centered gaussian stochastic process of variance σ^2, uncorrelated in time, and uncorrelated among individuals and among tasks" (Theraulaz et al. 1998, p. 328).

Implementation in Matlab

FTM

1. Initialization

At first, the parameter values are defined in the code (analogue Bonabeau et al. 1996).

```
clear all

%initial state
Theta_1=8; %Threshold caste 1
Theta_2=1; %Threshold caste 2
delta=1; %increase of stimulus per unit time
alpha=3; %a scale factor measuring the efficiency of task performance
N100=100;N1000=1000; %Population sizes
t_sim=1000; %simulation time
```

```
p=0.2; %probability that active individual becomes inactive again
```

2. Reproduction figure 1 in Bonabeau et al., 1996

The first figure in REF Bonabeau et al. 1996 (further referred to as Bonabeau_figure1) illustrates exemplarily the notion of response curves (`P1` and `P2`) with different response thresholds for majors (`T1`) and minors (`T2`), for stimulus intensities between 0.1 and 100 (`sti=[0.1:0.1:100]`). A logarithmic scale was chosen for displaying as a figure.

```
%Illustration of the notion of repose curves with different thresholds
%for minors and majors, Figure 1 (FTM, Bonabeau et al., 1996)
sti=[0.1:0.1:100]; %stimulus intensity
T1=8; T2=1; %Thresholds for caste 1 and 2
P1=sti.^2./(sti.^2+T1^2); %Probability for task performance caste 1
P2=sti.^2./(sti.^2+T2^2); %Probability for task performance caste 2
figure;
semilogx(sti,P1);hold on,semilogx(sti,P2,'r'),grid on
xlabel('Stimuli s'),ylabel('Probability P'),legend('Major','Minor')
title('Probability of engaging in task performance in response to simu-
lus')
```

3. Reproduction figure 2 in Bonabeau et al., 1996

With the formula set of the FTM, the “number of acts” per major (a name of a caste) can be calculated as a function of the share of the majors on total population. Figure 2 in Bonabeau (1996) (further referred to as Bonabeau_figure2) compares simulated results for two population sizes (N100=100 and N1000=1000) with experimental results shown in Wilson (1984).

The function “Number of acts” is implemented in a separate file `Nr_acts_np.m`:

```
function [N_k]=Nr_acts_np(k,N,delta,Theta_1,Theta_2,alpha,t_sim,p);

%---------------------------------------------------------------------
%This program calculates the number of acts per major during simulation
%as a function of the giving fraction of majors.
%Input parameters:
%k: Fraction of majors [0,1]
%N: Total number of potentially active individuals
%Theta_1: Initial threshold for the majors
%Theta_2: Initial threshold for the minors
%alpha: Scale factor measuring the efficiency of task performance
%t_sim: Simulation time
%p: Probability that an active individual gives up task performance and
%becomes inactive
%Output parameters:
%N_k: Number of acts per major during simulation for certain k
%---------------------------------------------------------------------


for i=1:length(k) %Simulation loop
    N1(1)=0;N2(1)=0;s(1)=0;P1(1)=0;P2(1)=0;
    for t=1:t_sim %Time loop
        s(t+1)=s(t)+delta-alpha*(N1(t)+N2(t))/N;
```

```
        P1(t+1)=s(t+1)^2/(s(t+1)^2+Theta_1^2);
        P2(t+1)=s(t+1)^2/(s(t+1)^2+Theta_2^2);
        qq=0;pp=0;
        for aa=1:k(i)*N %Agent loop 1
            agenta1(aa)=rand(1);
            if agenta1(aa)<P1(t+1)
                qq=qq+1;
            end
        end
        for bb=1:(1-k(i))*N %Agent loop 2
            agenta2(bb)=rand(1);
            if agenta2(bb)<P2(t+1)
                pp=pp+1;
            end
        end
        N1(t+1)=qq;N2(t+1)=pp; %saving act counts in vectors
    end
    N_k(i)=sum(N1)/(k(i)*N)*p; %Number of acts for each k
end
```

With the `Simulation loop`, different shares of majors on total populations (`k`) are considered. In the `Time loop`, `s`, `P1` and `P2` are calculated in discrete time steps. `P1` and `P2` relate to the probabilities of engaging in task performance for caste 1 and 2 (respectively majors and minors). The `Agent loop 1` is counting acts of individuals of caste 1 (`qq`), while `Agent loop 2` is counting those of caste 2 (`pp`). For each individual of a caste – that is for each loop `1:k(i)*N` respectively `1:(1-k(i))*N` – we pick a random number between 0 and 1 (`rand(1)`), and we only count an additional act of the caste, if this random number is below `P1` respectively `P2,` which were calculated in the `Time loop` before. The `Time loop` closes with saving the "act counts" in vectors `N1` (for caste 1) respectively `N2` for caste 2, as starting values for the next time loop and for subsequent analysis. The `Simulation loop` ends with calculating number of acts per major during simulation for each `k`. The first term relates to the number of acts of majors during simulation per number of total majors in population (sum(N1)/(k(i)*N)) and the second term accounts for the probability to give up a task (p). The second term can be understood as dividing the number of acts by the average time spent by an individual in task performance (1/p), and thus considers that the task performance stops after a certain amount of time. In this way the acts, which were performed by the same individual in 1/p time steps and which were counted repeatedly during the simulation, are subtracted.

Bonabeau_figure2 can then be reproduced by referring to this function. The input parameters used in the function are defined in the initialization section and retrieved by the sequence of their appearance in the related function term.

```
k=[0.1:0.1:1]; %Fraction of majors

[N_k100]=Nr_acts_np(k,N100,delta,Theta_1,Theta_2,alpha,t_sim,p);
[N_k1000]=Nr_acts_np(k,N1000,delta,Theta_1,Theta_2,alpha,t_sim,p);
```

```
figure;
plot(k,N_k100),hold on, plot(k,N_k1000,'r'),xlabel('Fraction of ma-
jors'),ylabel('Number of acts'),
title('Number of acts per major during simulation'),grid
on,legend('simulation N=100','simulation N=1000');
```

4. Reproduction figure 3 in Bonabeau et al., 1996

In Bonabeau_figure3, the time evolution of the number of majors engaged in task performance is shown with the rate of demand increase being multiplied by 2 at t=500 (`delta` = 1 → `delta` = 2). The fraction of majors is fixed and is equal to 0.2 (`k` =0.2). The code implemented in Matlab reads as follows:

```
clear k %Delete k from previous simulation
%Set simulation parameters,
k=0.2;
t_sim=1000;N=N100;
s(1)=0;
delta=1;


for t=1:t_sim %Time loop
    if t>499
        delta=2; %change delta beginning t=500
    end
        P1(t)=s(t)^2/(s(t)^2+Theta_1^2);
        P2(t)=s(t)^2/(s(t)^2+Theta_2^2);
        qq=0;pp=0;
        for aa=1:k*N %Agent loop 1
            agenta1(aa)=rand(1);
            if agenta1(aa)<P1(t)
                qq=qq+1;
            end
        end
        for bb=1:(1-k)*N %Agent loop 2
            agenta2(bb)=rand(1);
            if agenta2(bb)<P2(t)
                pp=pp+1;
            end
        end
        N1(t)=qq;N2(t)=pp; %saving act counts in vectors
        s(t+1)=s(t)+delta-alpha*(N1(t)+N2(t))/N; %Evolution of stimulus
        %intensity with time
end

for t=1:t_sim %simulation loop, sum of acts by majors
       if t<1/p
            sp1(t)=sum(N1(1:t));
       else
            sp1(t)=sum(N1(t-(1/p)+1:t));
       end
end

figure;
```

```
plot([1:t_sim],sp1(1:t_sim)),hold on,plot([500 500],[0 10],'r--
'),text(500,8,'increase of demand')
xlabel('Time steps'),ylabel('Number of majors'),title('Number of majors
involved in task performance'),grid on
```

The `Time loop` initially defines the moment in simulation, when `delta` is multiplied by two, then – concordant with reproduction of Bonabeau_figure 2, `P1` and `P2` are defined and the agent loops are implemented.

Subsequently, in the code the `simulation loop` is defined. The individuals which were active in the last `1/p` time steps are still involved in task performance and are counted in this loop. For the time `t<1/p,` the active individuals are counted from the first time step of the simulation respectively. For the time `t>1/p`, they are counted for the last `1/p` time steps.

5. Reproduction figure 4 in Bonabeau et al., 1996

In Bonabeau_figure4 the number of acts per major during simulation is calculated as function of fraction of majors in the population for different values of `z`, which is the fraction of the square of threshold from caste 1 and the square of threshold from caste 2 (Theta_1^2/Theta_2^2). This is implemented by varying `Theta_1` and keeping `Theta_2` as 1 (from previous definition) and using the function `Nr_acts_np.m` defined when reproducing Bonabeau_figure2.

```
Theta_1=[2 3 5 8 10];
delta=1;
k=[0.1:0.1:1]; %Fraction of majors
[N_z1]=Nr_acts_np(k,N100,delta,Theta_1(1),Theta_2,alpha,t_sim,p);
[N_z2]=Nr_acts_np(k,N100,delta,Theta_1(4),Theta_2,alpha,t_sim,p);
[N_z3]=Nr_acts_np(k,N100,delta,Theta_1(5),Theta_2,alpha,t_sim,p);

figure;
plot(k,N_z1),hold on,plot(k,N_z2,'r'),plot(k,N_z3,'g'),
xlabel('Fraction of majors'),ylabel('Number of acts'),title('Number of
acts per major during simulation')
legend('z=4','z=64','z=100')
```

VTM

1. Initialization

At first, the parameter values are defined in the code (analogue Theraulaz et al. 1998). Some are identically from the FTM introduced before, some are new.

```
Clear all

%Definition of the input parameters
delta=1; % Increase in stimulus intensity per unit time
N=5; %Number of individuals
```

```
m=2; %Number of tasks
alpha=3; %Scale factor measuring the efficiency of task performance
xi=10; % learning factor
p=0.2; Probability that an active individual gives up task performance
%and becomes inactive
phi=1 % forgetting factor
sigma=0.1; % Variance for a gaussian stochastic process
theta0=500*ones(N,m); %Initial response thresholds, same for all
individuals and all tasks
x0=0.1*ones(N,m); % Initial values for fraction of time spent in task
%perfomance, same for all individuals and all tasks
theta_max=1000;theta_min=0; %Restriction for thresholds
s0=0*ones(1,m); %Initial stimuli
```

2. Reproduction figure 1a) in Theraulaz et al., 1998

A program is written, which calculates the thresholds and fraction of time spent by an individual for task performance as time series. It is saved as `threshold.m` and describes a function, relating to formulas (4)-(7) in this report (formulas 1,4,5,7 in Theraulaz et al. 1998):

```
function [theta1 theta2 x1 x2]=threshold (del-
ta,N,m,alpha,p,xi,phi,sigma,theta0,x0,s0,theta_max,theta_min);

%-----------------------------------------------------------------------
%This programm calculates the Thresholds and fraction of time spent by
an
%individual for task performance as time series.
%Input parameters:
%delta: Increse in stimulus intensity per unit time
%N: Number of individuals
%m: Number of tasks
%alpha: Scale factor measuring the efficiency of task performance
%p: Probability that an active individual gives up task performance and
%becomes inactive
%xi: learning factor
%phi: forgetting factor
%sigma: Variance for a gaussian stochastic process
%theta0: Initial thresholds
%x0: Initial values for x
%s0: Initial stimuli
%theta_max,theta_min: Restriction for thresholds
%-----------------------------------------------------------------------

theta=theta0;s=s0;x=x0;

for t=1:3000 % time loop

    for i=1:N %agent loop, boundary definition formula 4 in Theraulaz
et al. 1998

        for j=1:m % task loop

            if (theta(i,j)-theta_min)>0
```

```
            faktor_min(i,j)=1;
            else faktor_min(i,j)=0;
            end
            if (theta_max-theta(i,j))>0
            faktor_max(i,j)=1;
            else faktor_max(i,j)=0;
            end
        end
    end
%formula 4 Theraulaz et al. 1998
delta_theta=((1-x)*phi-x*xi).*faktor_min.*faktor_max;

%formula 1 Theraulaz et al. 1998
s_temp=ones(N,1)*s;
T=s_temp.^2./(s_temp.^2+theta.^2);

%formula 5, Theraulaz et al. 1998
    for i=1:N
        sum_xi(i)=sum(x(i,:)); %"active time of individual i"
    end
sum_xij=sum_xi'*ones(1,m);
delta_x=T.*(1-sum_xij)-p*x+randn(N,m)*sigma;

%formula 7, Theraulaz et al. 1998
    for j=1:m
        sum_xj(j)=sum(x(:,j)); %"time spent on fulfilling task j of all
individuals"
    end

delta_s=delta-alpha/N*sum_xj;

%definition of starting values for next round
x=delta_x+x;
theta=delta_theta+theta;
s=s+delta_s;

%definition of output parameters
theta1(:,t)=theta(:,1);
theta2(:,t)=theta(:,2);
x1(:,t)=x(:,1);
x2(:,t)=x(:,2);

end
```

The `time loop` defines the simulation time and includes the `agent loop` and `task loop`, which implement the boundary conditions of the thresholds (the step function in formula (5) in this report). It follows, still within the time loop, the implementation of the formulas for `delta_theta` (response threshold dynamics, corresponding to formula (5) in this report), `T` (probability for task performance, corresponding to formula (4) in this report), `delta_x` (dynamics of time spent in task perfomance, corresponding to formula (6) in this report) and `delta_s` (dynamics of stimuli intensity, corresponding to formula (5) in this report). In order to define `T` stimuli intensity `s_temp` is calculated, which is the stimuli intensity at a specific time. For calculating `delta_x`, another agent loop is

implemented in order to calculate the “active time” of the individual (`sum_xi(i)`). Another task loop is implemented to calculate “time spent from all individuals for fulfilling a task” (`sum_xj(j)`), which is necessary to determine `delta_s`.
Output parameters are `theta1(:,t)`, which is the thresholds for task 1 (for all individuals at each time step), `theta2(:,t)`, which is the thresholds for task 2 (for all individuals at each time step), `x1(:,t)`, which is the fraction of time spent in performance of task 1 and `x2(:,t)`, which is the fraction of time spent in performance of task 2. The matrices `x` and `theta` have N rows and 3000 columns. By defining `theta1` and `2` and `x1` and `2` we extract the necessary information for generating the figure 1a) and figure 1b).

The first figure in Theraulaz et al. 1998 (further referred to as Theraulaz_figure1a) shows the dynamics of response thresholds for 5 individuals (`N=5`) and two tasks (`m=2`), with same initial thresholds for all individuals and tasks (`theta0=500*ones(N,m)`) It is reproduced by using the following code:

```
for i=1:10000
%reference to function
[theta1 theta2 x1 x2]=threshold (del-
ta,N,m,alpha,p,xi,phi,sigma,theta0,x0,s0,theta_max,theta_min);

if (theta1>-100) & (theta2>-100) %stop criterion
    break;
end
end

zeichen1={'--b.','--g.','--r.','--y.','--k.'};
zeichen2={'--b+','--g+','--r+','--y+','--k+'};

%figure 1a) Theraulaz et al. 1998
figure;
for i=1:N
    plot(theta1(i,:),zeichen1{i},'MarkerSize',3),hold on
    plot(theta2(i,:),zeichen2{i},'MarkerSize',3),hold on
end
le-
gend('theta11','theta12','theta21','theta22','theta31','theta32','theta
41','theta42','theta51','theta52')
xlabel('Time'),ylabel('Theta'),title('Theta with 5 individuals and 2
tasks'),grid on
```

The `stop criterion` for the calculation of the thresholds (`theta1`, `theta2`) is defined in order to avoid illogical results, which could occur due to the random process in the formula for `delta_x` (`fractions of time spent in task performance`, see `threshold.m`).

3. Reproduction figure 1b) in Theraulaz et al., 1998

For reproducing Theraulaz_figure1b), the same function as introduced for Theraulaz_figure1a) is referred to, the initial values remain unchanged and the stop criterion is active. Only the plot instruction is changed, since here the dynamics of the fraction of time spent by individual `i` in performing task 1 (`x1`) and task 2 (`x2`) shall be illustrated:

```
%figure 2b) Theraulaz et al. 1998
figure;
for i=1:N
    plot(x1(i,:),zeichen1{i},'MarkerSize',3),hold on
    plot(x2(i,:),zeichen2{i},'MarkerSize',3),hold on
end
le-
gend('theta11','theta21','theta31','theta41','theta51','theta12','theta
22','theta32','theta42','theta52')
xlabel('Time'),ylabel('x'),title('Fraction of time spent in task per-
formance'),grid on
```

4. Reproduction figure 1c) in Theraulaz et al., 1998

Also for the reproduction of Theraulaz_figure1c) the proceeding is similar to 1a) and 1b). This figure is identical to Theraulaz_figure1a) despite the fact that "the initial distribution of thresholds is uniform over [θ_{min} =1 and θ_{max}.=1000]" (Theraulaz et al. 1998). This implies that `theta0` is newly defined and the calculation needs to be implemented again.

```
%figure 2c) Theraulaz et al. 1998
%Definition of the initial threshold (as defined in paper)
theta0=[150 50;900 350;450 170;880 200;650 800];

for i=1:10000
[theta1 theta2 x1 x2]=threshold (del-
ta,N,m,alpha,p,xi,phi,sigma,theta0,x0,s0,theta_max,theta_min);
if (theta1>-100) & (theta2>-100)
    break; end
end

figure;
for i=1:N
    plot(theta1(i,:),zeichen1{i},'MarkerSize',3),hold on
    plot(theta2(i,:),zeichen2{i},'MarkerSize',3),hold on
end
le-
gend('theta11','theta12','theta21','theta22','theta31','theta32','theta
41','theta42','theta51','theta52')
xlabel('Time'),ylabel('Theta'),title('Theta with different initial val-
ues'),grid on
```

Adaptation

The VTM model is further tested with choosing different values for the parameters that are stating the learning and forgetting rate (ξ and φ). We reproduce figures 1a) and 1c) – that is the dynamics of response thresholds for initially identical thresholds and for initially differentiated thresholds – for various ξ and φ, while their sum stays constant (ξ

+ φ=11).
We then define a “minimal learning factor”, which guarantees a division of labour and thus assures that some individuals stay active. This implies that their individual thresholds are low enough that activity can possibly be observed. .

We graphically and statistically show turning points for both cases, that is, for which values a division of labour occurs and for which values no specification is observed in the simulation.

The code to show graphically the effects of varying phi and xi *(ξ(10:-1:5) and φ(1:1:6))* reads as follows:

```
%Definition of the initial values
step=1; %time step
delta=1;N=5;m=2;alpha=3;xi=10;p=0.2;Phi=[1:step:6];sigma=0.1;theta0=500
*ones(N,m);x0=0.1*ones(N,m);theta_max=1000;theta_min=0;s0=0*ones(1,m);

for ite=1:length(Phi)%iteration of phi
    phi=Phi(ite);
    theta0=500*ones(N,m); %because theta0 is newly defined for second
figure
for i=1:10000
[theta1 theta2 x1 x2]=threshold
(delta,N,m,alpha,p,xi,phi,sigma,theta0,x0,s0,theta_max,theta_min);
if (theta1>-100) & (theta2>-100)
    break; end
end


zeichen1={'--b.','--g.','--r.','--y.','--k.'};
zeichen2={'--b+','--g+','--r+','--y+','--k+'};

%figure 1a) Theraulaz et al. 1998, reproduced with different phi
figure(1);
for i=1:N
    subplot(2,3,ite),plot(theta1(i,:),zeichen1{i},'MarkerSize',3),hold
on
    subplot(2,3,ite),plot(theta2(i,:),zeichen2{i},'MarkerSize',3),hold
on
end
xlabel('Time'),ylabel('Theta'),title(['Theta with 5 individuals and 2
tasks with phi=',num2str(Phi(ite))]),grid on

%figure 2c) Theraulaz et al. 1998
%Definition of the initial thresholds
theta0=[150 50;900 350;450 170;880 200;650 800];

%Fixed thresholds with different initial thresholds
for i=1:10000
[theta1 theta2 x1 x2]=threshold (del-
ta,N,m,alpha,p,xi,phi,sigma,theta0,x0,s0,theta_max,theta_min);
if (theta1>-100) & (theta2>-100)
    break; end
end
```

```
figure(2);
for i=1:N
    subplot(2,3,ite),plot(theta1(i,:),zeichen1{i},'MarkerSize',3),hold
on
    subplot(2,3,ite),plot(theta2(i,:),zeichen2{i},'MarkerSize',3),hold
on
end
%le-
gend('theta11','theta12','theta21','theta22','theta31','theta32','theta
41','theta42','theta51','theta52')
xlabel('Time'),ylabel('Theta'),title(['Theta with different initial
values with phi=' num2str(Phi(ite))]),grid on

xi=xi-step;
end
fig-
ure(1);subplot(2,3,6),legend('theta11','theta12','theta21','theta22','t
heta31','theta32','theta41','theta42','theta51','theta52')
fig-
ure(2);subplot(2,3,6),legend('theta11','theta12','theta21','theta22','t
heta31','theta32','theta41','theta42','theta51','theta52')
```

A second program, implemented in order to do statistics to find the “minimal learning factor”, calculates the number of the performed tasks with a smaller time step of phi and xi (ξ(10.6:-0.2:5), φ(0.4:0.2:6)) again for two cases: One with identical initial thresholds for all individuals (similar Theraulaz_figure1a) and the other with different initial thresholds (similar Theraulaz_figure1b). The time step of `phi` is defined as 0.2 and the thresholds smaller 0.5 are counted as tendency for task performance. The minimal learning factor is defined as xi2 in the program and initially set as 0. After running this program, one can find the minimal learning factor, which guarantees that at least one individual has a low threshold and is ready for task performance. It can then be set manually by replacing 0. Implicitly this also sets a “maximum forgetting rate” the forgetting rate (since they are coupled by a constant sum).
We have to note that with a `phi` smaller than 0.4, we have to wait for a very long time to get a logical result. Therefore, our test starts with 0.4 as minimal value of `phi`.

```
clear all
%For statistics with different learning and forgetting rates.
%Learning+Forgetting=11!

%Definition of the initial values
step=0.2;
delta=1;N=5;m=2;alpha=3;p=0.2;
xi2=0; %Minimal learning factor, initially set as 0, can be varied
manually
Phi=[0.4:step:6]; %Forgetting rate
xi =11-Phi(1); %learning rate
sigma=0.1;theta0=500*ones(N,m);x0=0.1*ones(N,m);theta_max=1000;theta_mi
n=0;s0=0*ones(1,m);

%Identical initial threshold for all the agents and tasks
```

```
for ite=1:length(Phi)
    phi=Phi(ite);
    if phi>11- xi2 %stop function for phi (using minimal learning
factor) cannot be bigger than 11-xi2

        break;
    end
    theta0=500*ones(N,m);
for i=1:10000
[theta1 theta2 x1 x2]=threshold (delta,N,m,alpha,p,
xi,phi,sigma,theta0,x0,s0,theta_max,theta_min);
if (theta1>-100) & (theta2>-100)
    break; end
end

%count final thresholds below 0.5 (indicates task performance)
I1=length(find(theta1(:,3000)<0.5));I2=length(find(theta2(:,3000)<0.5))
;
I(ite)=I1+I2;

%figure 2c) Theraulaz et al. 1998
%Different initial threshold for all the agents and tasks
%Definition of the initial thresholds
theta0=[150 50;900 350;450 170;880 200;650 800];

for i=1:10000
[theta1 theta2 x1 x2]=threshold
(delta,N,m,alpha,p,xi,phi,sigma,theta0,x0,s0,theta_max,theta_min);
if (theta1>-100) & (theta2>-100)
    break; end
end

%count final thresholds below 0.5 (indicates task performance)
I1=length(find(theta1(:,3000)<0.5));I2=length(find(theta2(:,3000)<0.5))
;
II(ite)=I1+I2;

xi = xi -step; %set xi for next round

end
```

In order to generate a table showing the number of individuals involved in task performance with varying phi for the two cases (identical and different initial thresholds), the variables I and II in the code are displayed.

Simulation Results and Discussion

FTM

1. Reproduction figure 1 in Bonabeau et al., 1996

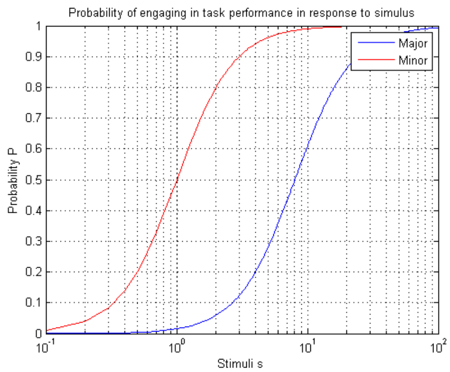

Figure 1: Bonabeau_figure1, reproduction in Matlab

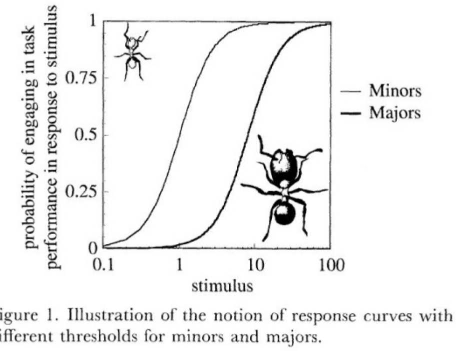

Figure 2: Bonabeau_figure1, copy from Bonabeau et al. 1996

Bonabeau_figure1 graphically shows that the probability to perform a task is rising with rising stimuli and that – due to their higher threshold – majors have lower response probabilities for similar stimuli than minors. For very high stimuli the response threshold θ becomes less important and probabilities for task performance both tend towards 100% task performance. The reproduction in Matlab (Figure 1) and the original graph in Bonabeau et al. 1996 (Figure 2) show similar results.

2. Reproduction figure 2 in Bonabeau et al., 1996

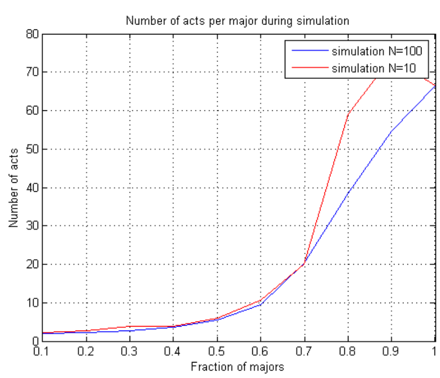

Figure 3: Bonabeau_figure2, reproduction in Matlab

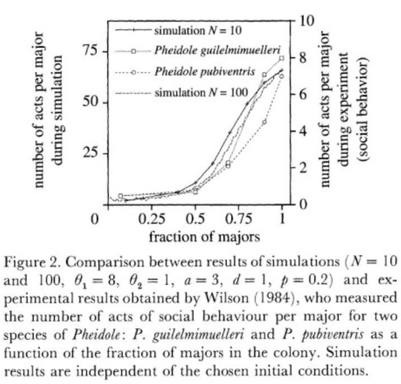

Figure 4: Bonabeau_figure2, copy from Bonabeau et al. 1996

In Bonabeau_figure2 (Figure 4), simulation results for numbers of acts of majors with different shares of majors in population are compared with experimental observations with two ant species. The graph shows that with increasing share of majors in population, the more active they are in task performance. This is due to the fact that fewer minors with lower response thresholds are available to fulfil tasks, which would reduce stimuli intensity. Therefore stimuli intensity is more likely to reach the higher threshold of majors which increases the probability of task performance and thus highlights the behavioural flexibility of workers in the ant colony.
A high qualitative overlap can be seen, independent of population size assumed in the

simulation (N=10 and N=100 show similar patterns). Compared to our reproduction in Matlab, shown in Figure 3, the same pattern can be observed; however, the very small population size (N=10) results here in a “illogical” sharp bent for high fraction of majors. Possibly the authors also used higher population sizes for this simulations, because with population sizes N=100 and N=1000 we receive a similar pattern.

3. Reproduction figure 3 in Bonabeau et al., 1996

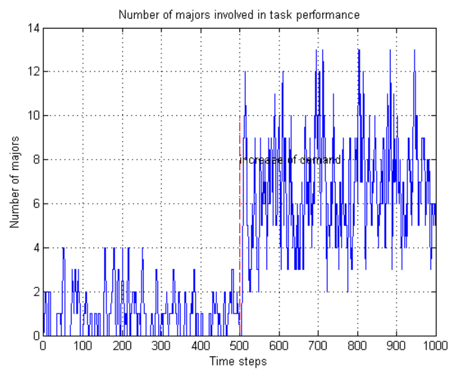

Figure 5: Bonabeau_figure3, reproduction in Matlab

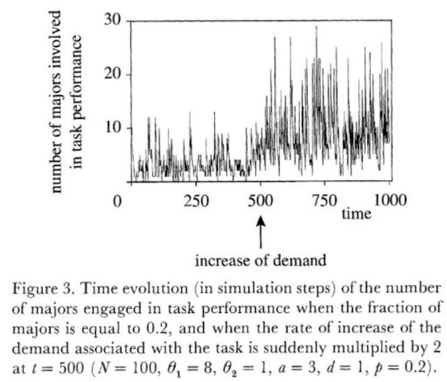

Figure 6: Bonabeau_figure3, copy from Bonabeau et al. 1996

The next figure in Bonabeau et al. 1996 further accentuates behavioural flexibility of the colony by showing how the number of majors involved in task performance increases when the demand is suddenly rising by a factor of 2. “most minors are already involved in task performance before this change, so that the involvement of majors is required to maintain the demand at a low enough level” (Bonabeau et al. 1996, p. 1567). The pattern of our reproduction in Matlab (Figure 5) is similar to the one in the original paper (Figure 6), however, scales vary by about a factor two. The reason for that is unclear.

4. Reproduction figure 4 in Bonabeau et al., 1996

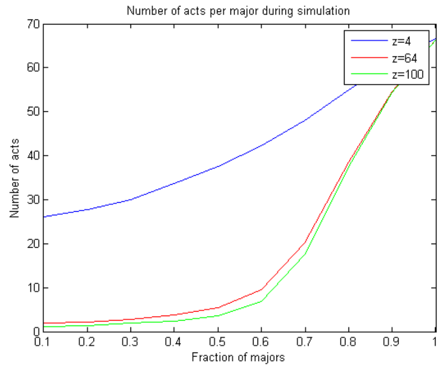

Figure 7: Bonabeau_figure4, reproduction in Matlab

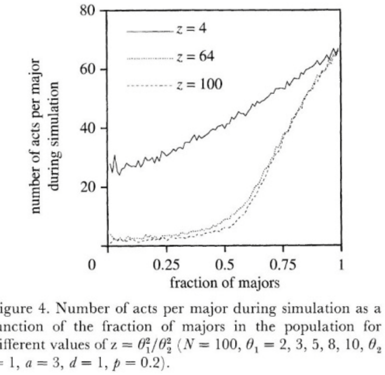

Figure 8: Bonabeau_figure4, copy from Bonabeau et al. 1996

Bonabeau_figure4 (Figure 8) shows variation in the shape of the curve as seen in Bonabeau_figure2 (Figure 4), if the ratio of the thresholds of both castes (Theta_1^2/Theta_2^2) is changed. "The transition becomes more abrupt [with increasing ratio], and the point at which this transition takes place decreases and seems to converge towards a limit (around 0.5)" (Bonabeau et al. 1996, p. 1567/8). With lower ratios, more majors are active in task performance, which can be understood from the convergence of Theta_1 (threshold majors) towards Theta_2 (threshold minors), which implies a convergence of activities of both castes.
Again, our simulation in Matlab (Figure 7) and the one from the original paper show similar patterns, however, the fluctuation is lower in our simulation (possibly due to longer time steps in our simulation than in the original paper).

VTM

1. Reproduction figure 1a) in Theraulaz et al., 1998

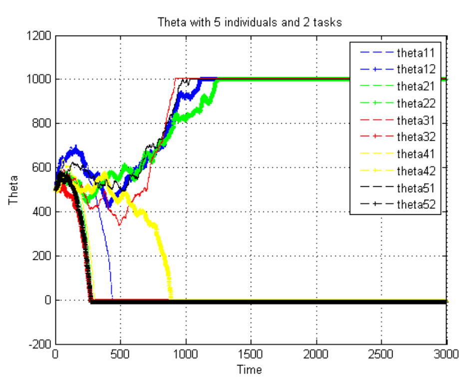

Figure 9: Theraulaz_figure1a), reproduction in Matlab

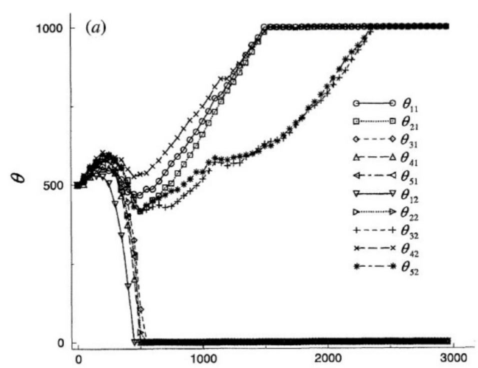

Figure 10: Theraulaz_figure1a), copy from Theraulaz et al. 1998

Theraulaz_figure1a) shows the dynamics of response thresholds θ_{ij} (Figure 10). In Theraulaz et al. (1998) it states: “A low value of θ_{ij}, indicates that individual *i* is highly sensitive to task *j*-associated stimuli and is therefore a specialist of task *j*. Individuals 3, 4 and 5 are task 1 specialists, and individuals 1 and 2 are task 2 specialists”. The figure highlights, that even though all individuals are starting with the same thresholds for both tasks, after a sufficient amount of time a division of labor occurs, only induced by the stochastic term in formula (6).

Our simulation results (Figure 9) shows similar patterns to the original figure in the paper.

2. Reproduction figure 1b) in Theraulaz et al., 1998

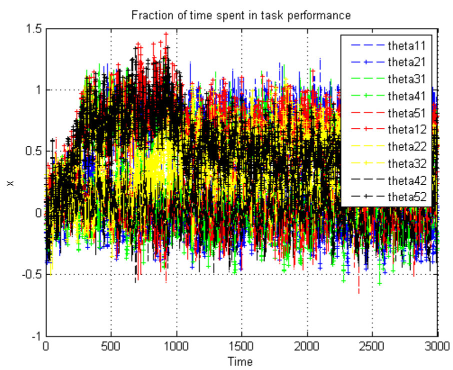

Figure 11: Theraulaz_figure1b), reproduction in Matlab

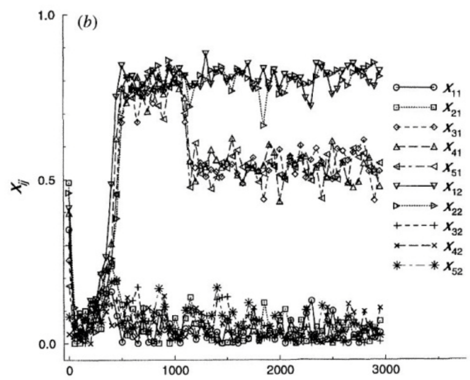

Figure 12: Theraulaz_figure1b), copy from Theraulaz et al. 1998

In Theraulaz_figure1b) same input parameters are used in order to calculate the dynamics of the fraction of time spent of individuals in task performance, which is expressed as x_{ij}, when *i* is referring to the individual and *j* to the task (Figure 12). In the caption of this figure it is further stated in Theraulaz et al. 1998: “When x_{ij} is close to 1, individual *i* spends most of its time performing task *j*. Individuals 3, 4 and 5, who all perform mostly task 1, are less active ($x_{i1} \approx 0.05$, $x_{i2} \approx 0.05$) than individuals 1 and 2, who perform mostly task 2 ($x_{i1} \approx 0.55$, $x_{i2} \approx 0.8$)”. Our simulation (Figure 11) shows a considerably higher variation and no clustering, the reason will be the definition of the stochastic term.

3. Reproduction figure 1c) in Theraulaz et al., 1998

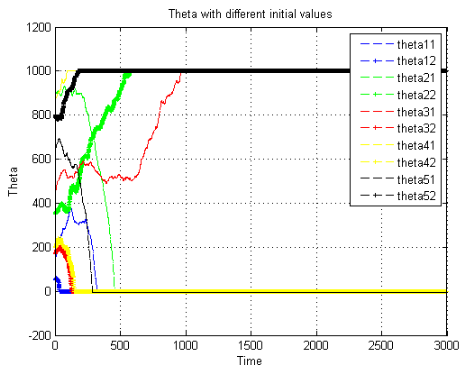

Figure 13: Theraulaz_figure1c), reproduction in Matlab

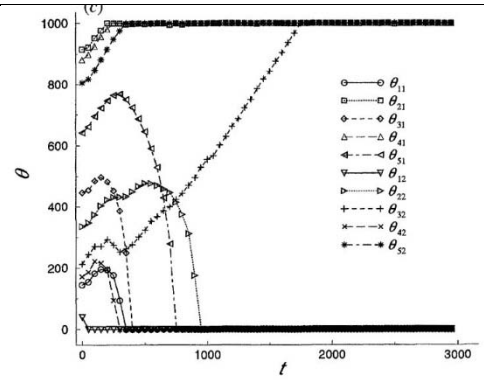

Figure 14: Theraulaz_figure1c), copy from Theraulaz et al. 1998

Theraulaz_figure1c) (Figure 13) is similar to Theraulaz_figure1a) (Figure 10), except that here the initial values for the response thresholds are different for all individuals and for all tasks. As for similar initial values, a division of labour is observed, but low initial values of an individual for a specific task make it more likely that this individual becomes specialist of this task (that is, that this thresholds tends towards 0). The caption of this figure in Theraulaz et al. 1998 further reads: “Individuals 1, 3 and 5 are task 1 specialists, and individuals 1, 2 and 4 are task 2 specialists (individual l is a specialist of both tasks)”. In reality it can be interpreted that the genotypic characteristic of an individual can predispose it to perform certain tasks. Our reproduction (Figure 13) is similar, however, the predestination is less pronounced, as for individual 2, despite the lower initial threshold for task 2 it becomes a task 1 specialist.

Adaptation

We graphically (ξ(10:-1:5), φ(1:1:6)) and statistically(ξ(10.6:-0.2:5), φ(0.4:0.2:6)) show the resulting thresholds for all individuals and all tasks with different learning and forgetting rates, for similar initial thresholds (similar Theraulaz_figure1a), see Figure 15) and for initially distributed thresholds (similar Theraulaz_figure1c), see Figure 16).

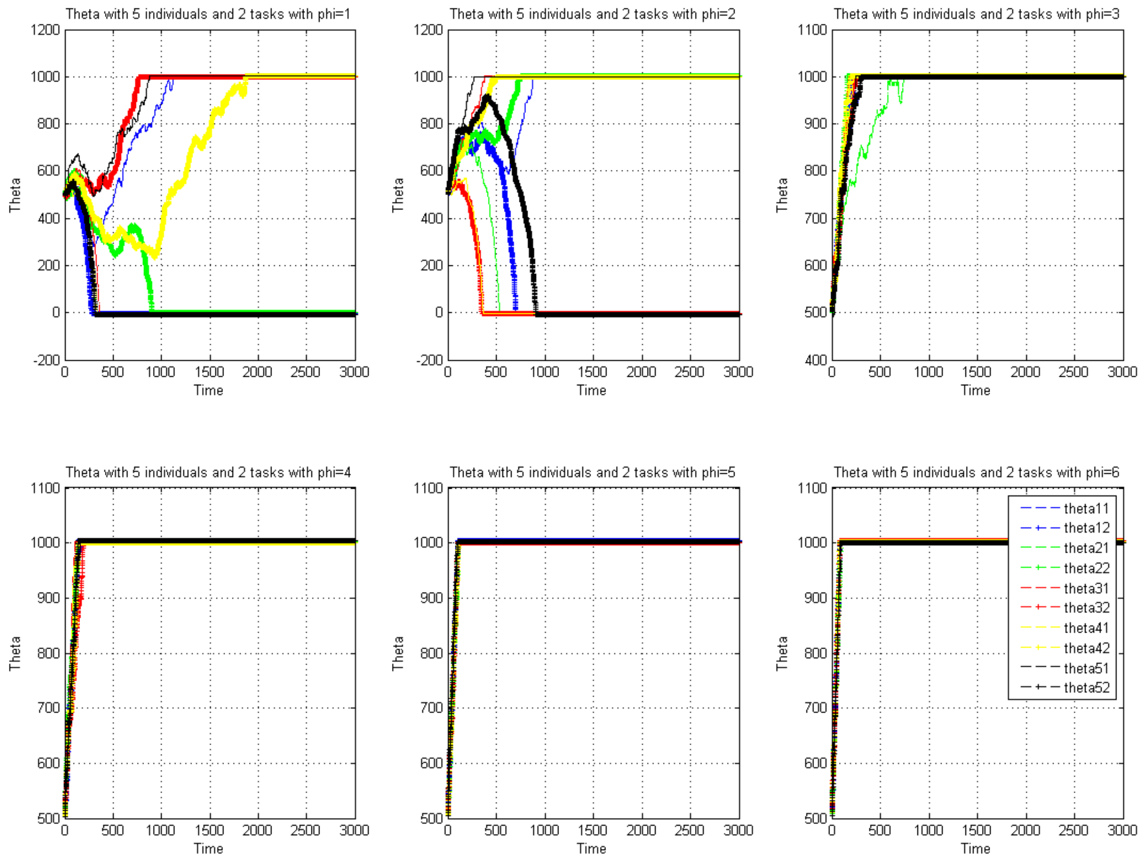

Figure 15: Theraulaz_figure1a) (similar initial thresholds), reproduction in Matlab, with varying phi (forgetting rate) and xi (learning rate)

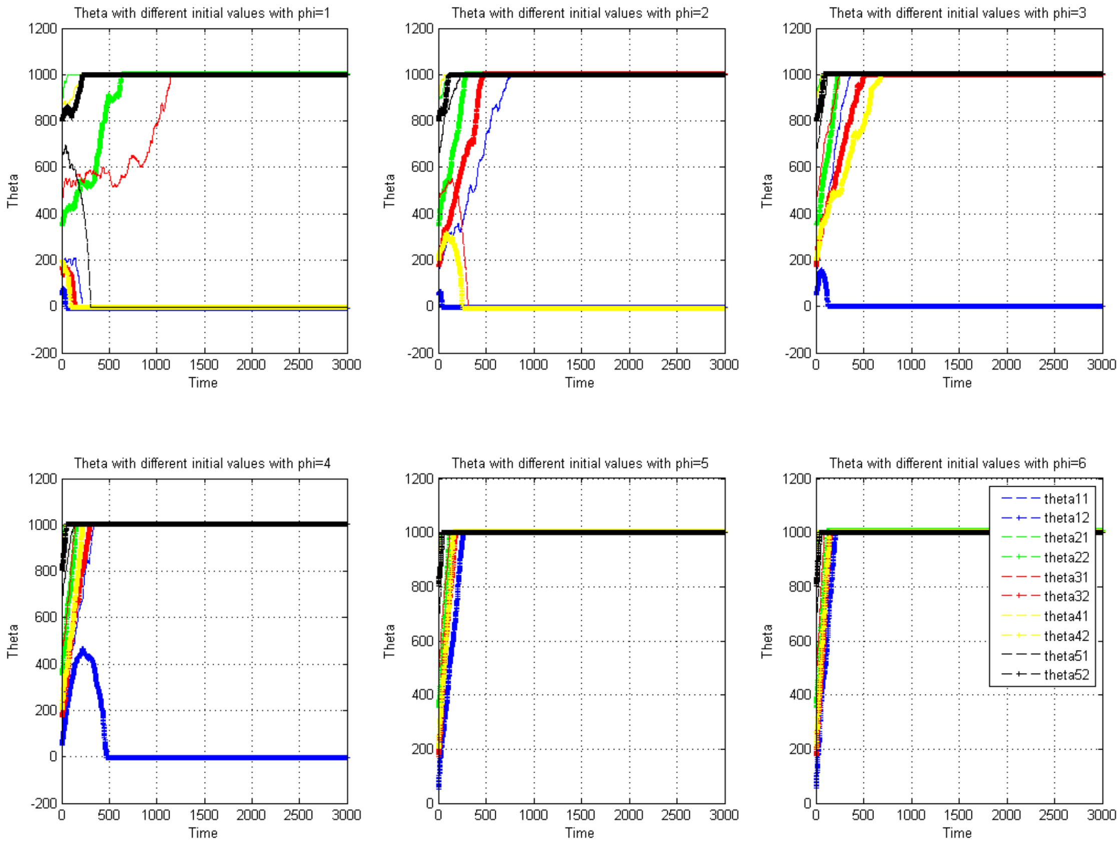

Figure 16: Theraulaz_figure1c) (different initial thresholds), reproduction in Matlab, with varying phi (forgetting rate) and xi (learning rate)

Figure 15 shows that the turning point for the division of labour happens at about phi=3 for the case with identical initial thresholds. With a higher forgetting rate the response thresholds of the individuals tend for all tasks to a maximum level; i.e. they stop task performance. It is also interesting to see that the turning point comes later for the case with different initial thresholds (see Figure 16) and that here individuals with smaller initial thresholds are more likely to become active in task performance.

Table 1 shows the resulting thresholds with a variation of phi in smaller steps. The turning point for the case with identical initial thresholds appears at phi=2.8, while it doesn't appear for the case with different initial thresholds until phi=4.2.
The minimal learning factor therefore needs to be set as xi= 8.2 for the case with identical initial thresholds (and xi= 6.8 for different initial thresholds), in order to guarantee division of labour (respectively that at least one individual stays active in task performance).

Table 1: Response thresholds with varying forgetting rates phi

φ	Threshold (identical phi0)	Threshold (different phi0)	φ	Threshold (identical phi0)	Threshold (different phi0)
0.4	6	6	0.6	7	7
0.8	6	6	1.0	5	6
1.2	5	5	1.4	8	5
1.6	5	5	1.8	5	5
2.0	5	3	2.2	4	2
2.4	4	3	2.6	2	2
2.8	1	1	3.0	0	1
3.2	0	1	3.4	0	1
3.6	0	1	3.8	0	1
4.0	0	1	4.2	0	1
4.4	0	0	4.6	0	0
4.8	0	0	5.0	0	0
5.2	0	0	5.4	0	0
5.6	0	0	5.8	0	0
6.0	0	0			

Summary and Outlook

With reproducing some key figures of the FTM and VTM model, we showed how the simple model (FTM) can be refined in order to account for learning and forgetting, and therewith for the genesis of task-allocation and within-caste specialization. We further proofed that our simulations in Matlab have similar patterns, but that the definition of the stochastic term, which was implemented differently in our model than in the original paper, influences the fluctuations in the model.

With our adaptations regarding the learning and forgetting factor, we showed that the model behavior, respectively the occurrence of division of labour, depends on the parameter definitions, e.g. the learning and forgetting factors. We showed that in order to observe a division of labor, respectively in order to see at least one individual performing a task, the “minimal learning factor” is 8.2 for the case of identical initial thresholds, and 6.8 for the case of different initial thresholds (with learning and forgetting factors coupled by their sum being always 11). Possible reasons for this observation are:

- We see a limitation of the model (artifact): It cannot account for extreme values (which could be present in nature). In this case, it would be necessary to implement a term in the model, which allows for specialization even though forgetting rates are high and learning rates are low.

- We reproduce a natural pattern: there is a minimal learning factor, otherwise the population is “too stupid”/absent-minded to develop division of labor. It is questionable, though, that this would result in a general “strike” of all individuals in the colony, as we observe with the trend towards high response thresholds (which can be translated as refusing to perform the task)

It would be necessary to cross-check with reality, which reason is more likely.

References

Bonabeau, E., G. Theraulaz and J.-L. Deneubourg (1996). "Quantitative Study of the Fixed Threshold Model for the Regulation of Division of Labour in Insect Societies." Proceedings of the Royal Society of London. Series B: Biological Sciences **263**(1376): 1565-1569.

Calabi, P. (1988). "Behavioral flexibility in Hymenoptera: a reexamination of the concept of caste." Advances in Myrmecology: 237-258.

Robinson, G. E. (1987a). "Modulation of alarm pheromone perception in the honey bee: evidence for division of labour based on hormonally regulated response thresholds." J. comb. Physiol. **160**(5): 613-619.

Robinson, G. E. (1987b). "Regulation of honey bee age polyethism by juvenile hormone." Behav. Ecol. Sociobiol **20**: 329-338.

Robinson, G. E. (1992). "Regulation of division of labour in insect societies." A. Rev. Entomol **37**: 637-665.

Theraulaz, G., E. Bonabeau and J.-N. Denuebourg (1998). "Response threshold reinforcements and division of labour in insect societies." Proceedings of the Royal Society of London. Series B: Biological Sciences **265**(1393): 327-332.