FROM PETRI NETS TO PARTIAL DIFFERENTIAL EQUATIONS

- SPATIAL MODELLING IN SYSTEMS BIOLOGY -

MONIKA HEINER

BRANDENBURG TECHNICAL UNIVERSITY COTTBUS-SENFTENBERG
COMPUTER SCIENCE INSTITUTE

. . . AND THEN THERE WAS COLOUR

Kew Gardens, 24/04/2011

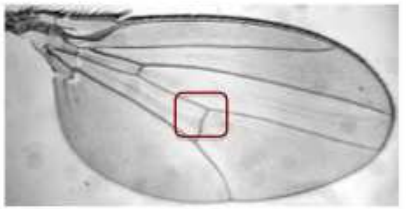
monika.heiner@b-tu.de

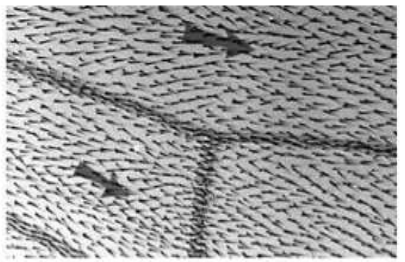
COLOUR -

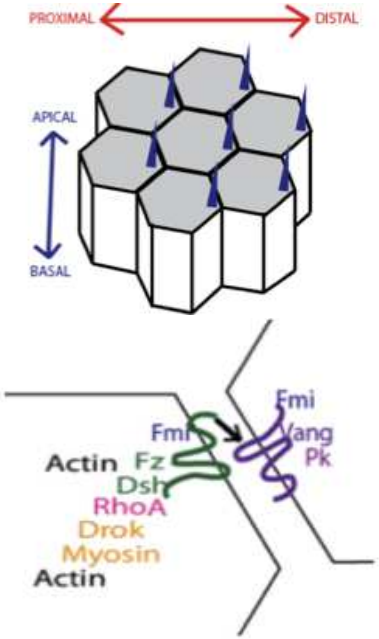
WHAT FOR ?

OUTLINE

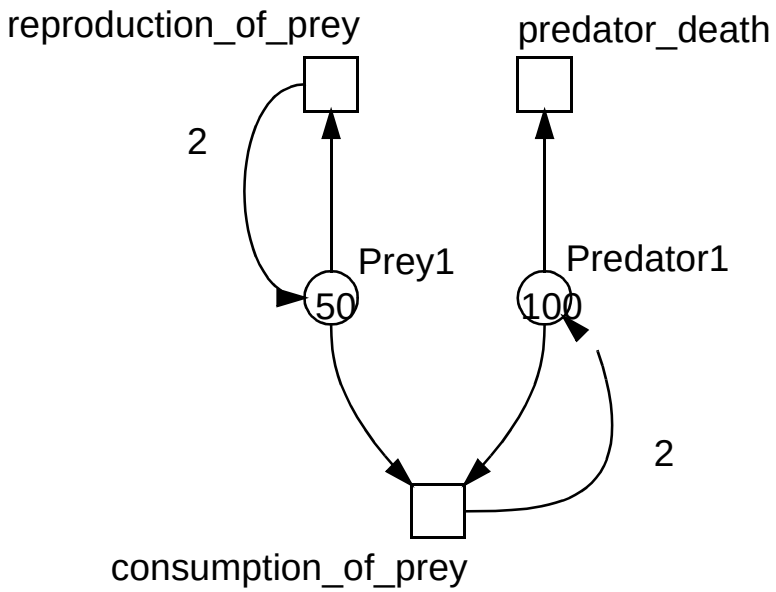
- **COLOUR, WHAT FOR ?**
  - -> *(bio) processes evolving in time and space*
  - -> *How to encode space ?*
- **COLOURING SPACE (VERSION 1)**
  - -> *diffusion in space*
  - -> *Turing patterns*
  - -> *phase variation in multistrain bacterial colonies*
  - -> *planar cell polarity in fly wing*
- **LIMITATION -> VERSION 2**
- **SUMMARY & OUTLOOK**
  - -> *next steps*



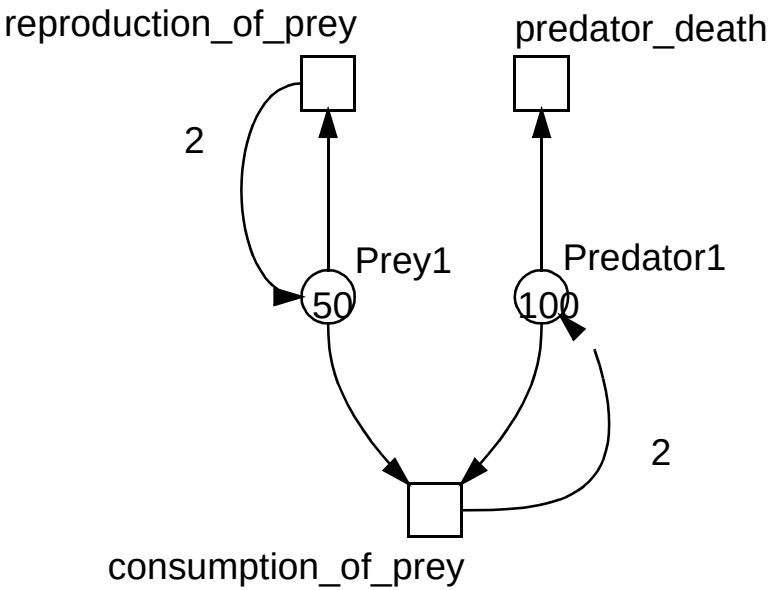




EX: PREY - PREDATOR

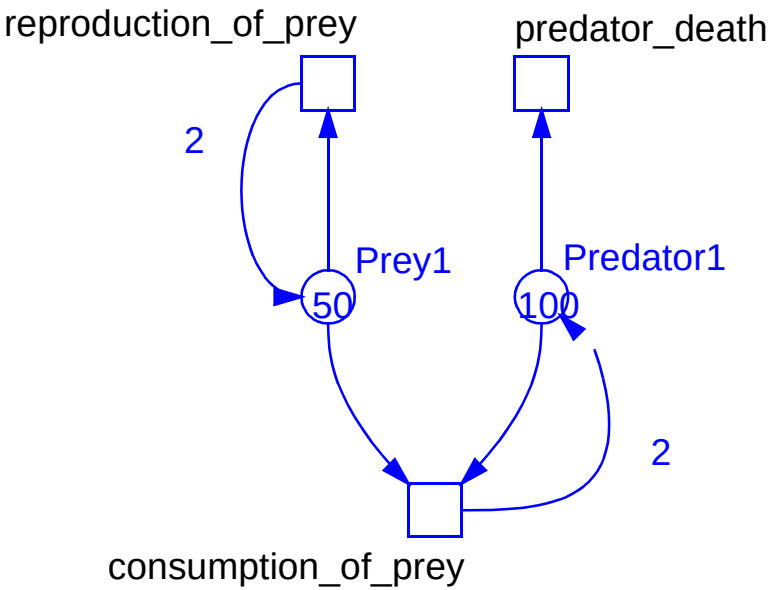


EX: PREY - PREDATOR

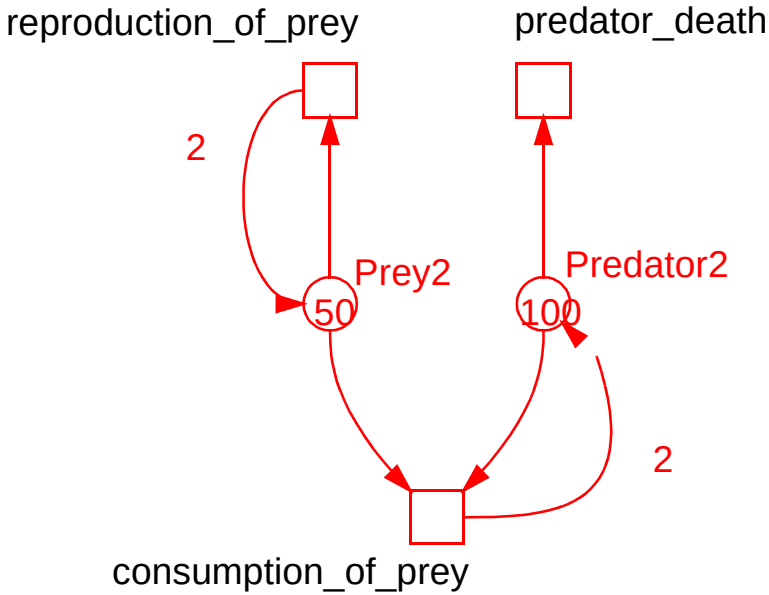


EX: PREY - PREDATOR

sub-system1



sub-system2



EX: PREY - PREDATOR

- **definitions**

  colourset *CS = 1-2;*

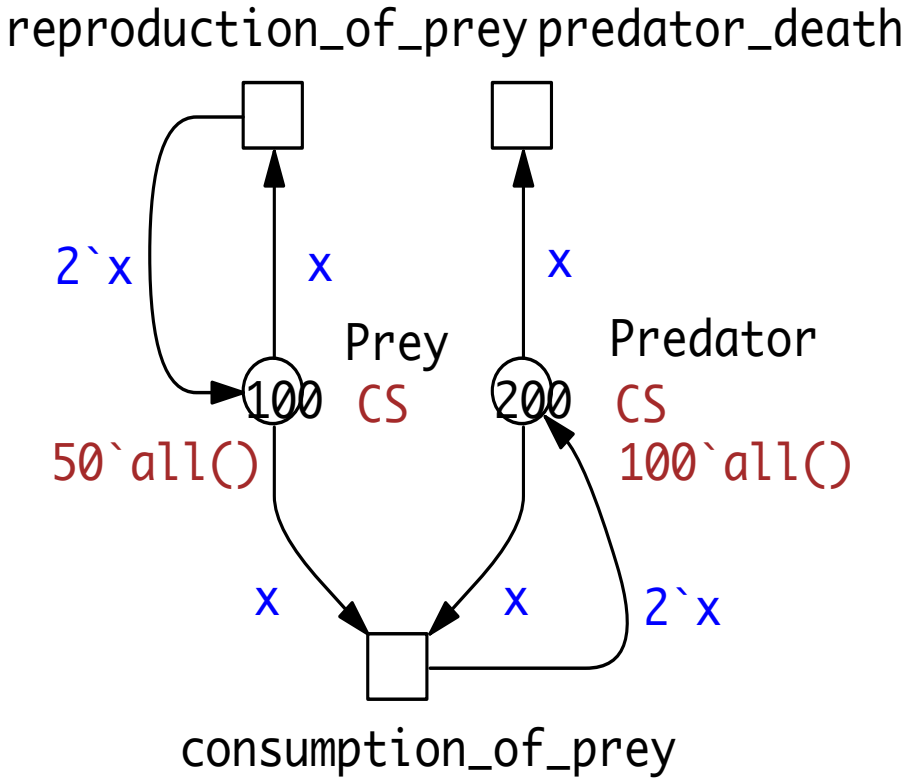
  var *x : CS;*

- **better:**

  const *SIZE = 2;*

  colourset *CS = 1-SIZE;*

  var *x : CS;*



EX: PREY - PREDATOR

- **definitions**

  colourset *CS = 1-2;*

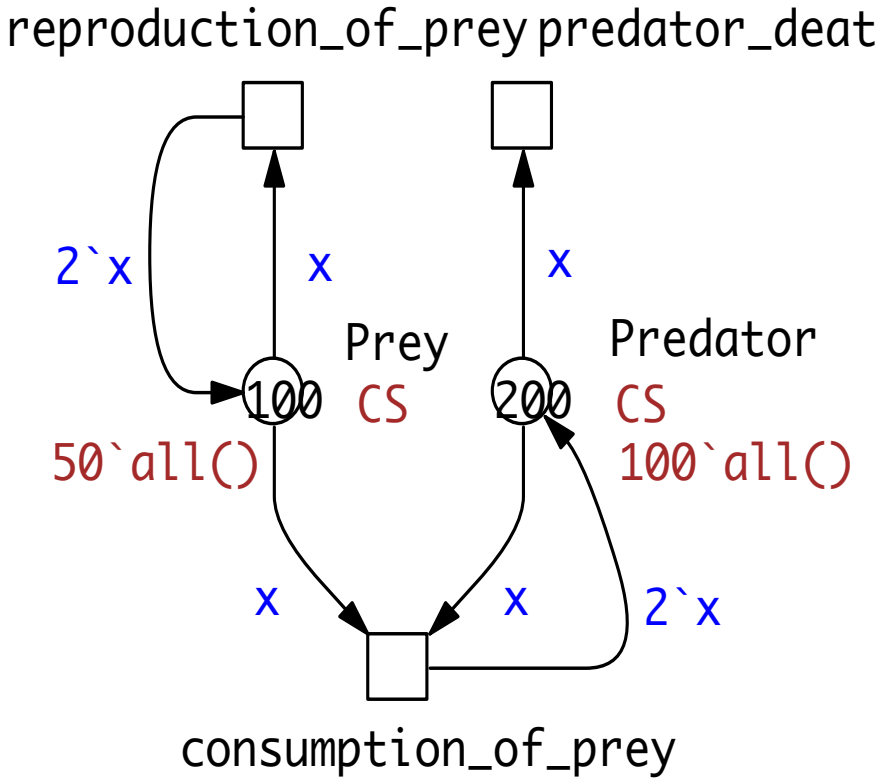
  var *x : CS;*

- **better:**

  const *SIZE = 2;*

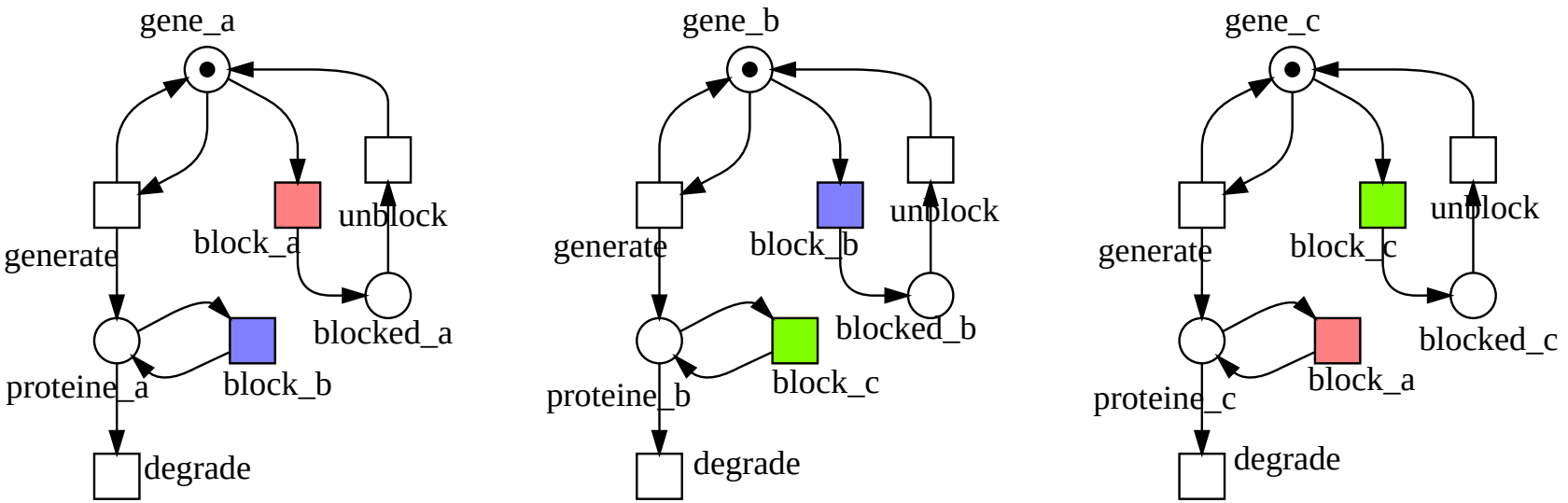
  colourset *CS = 1-SIZE;*

  var *x : CS;*

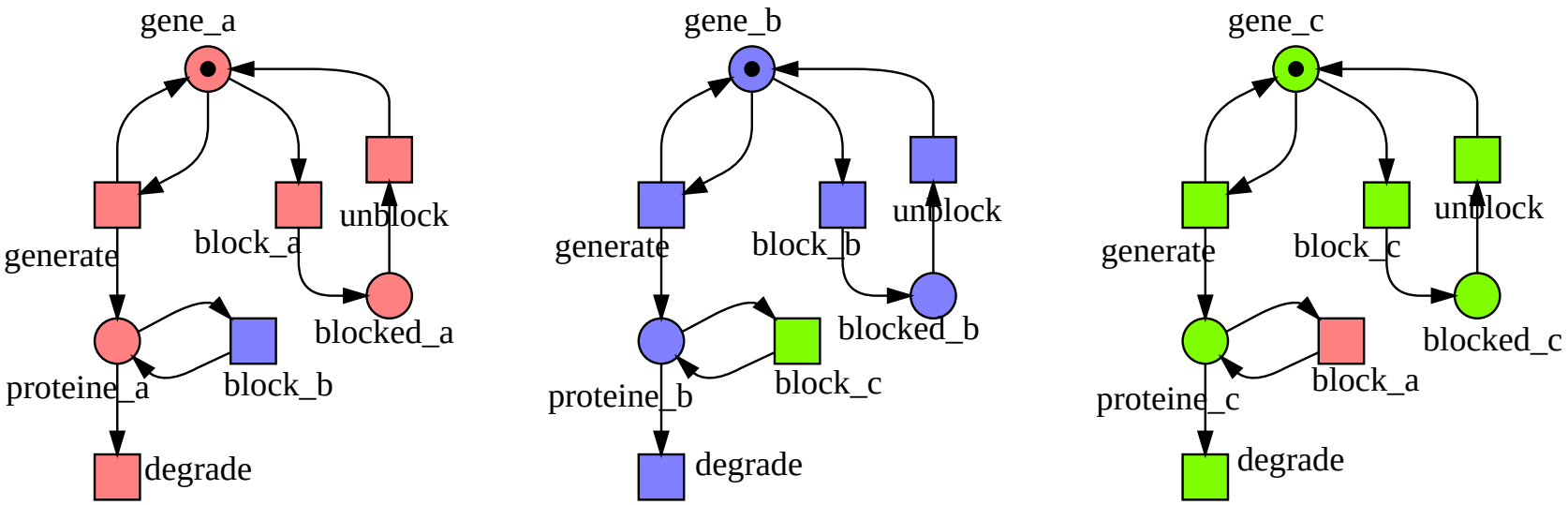


- **changing SIZE adapts the model to various scenarious**

EX: REPRESSILATOR



EX: REPRESSILATOR

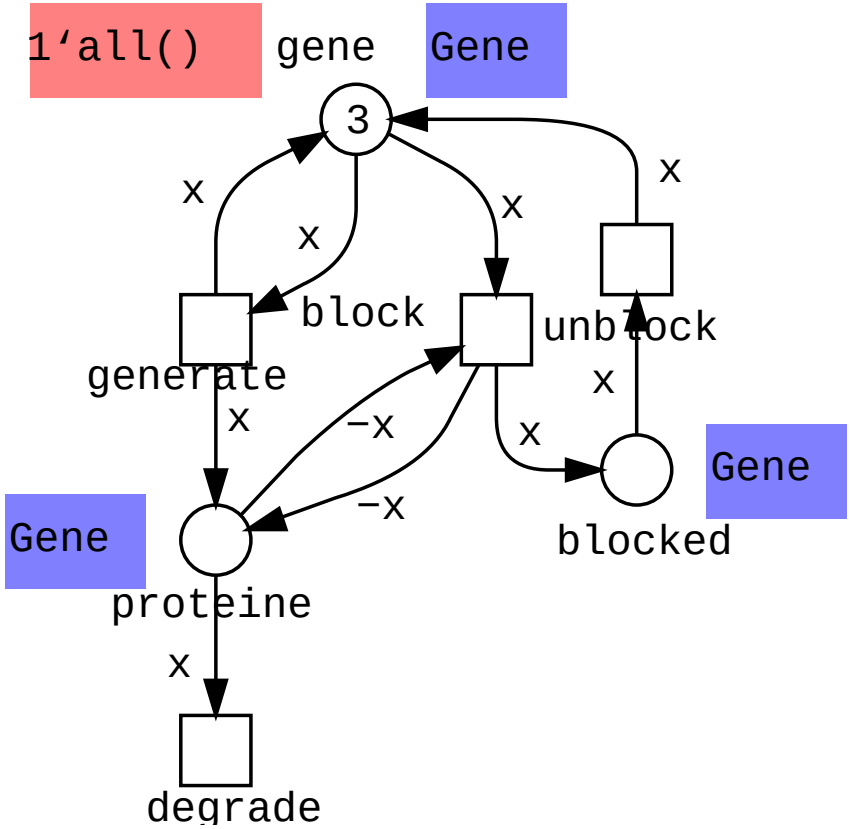


EX: REPRESSILATOR

- **definitions**

  colorset *Gene = enum a-c;*

  var *x : Gene;*



EX: REPRESSILATOR

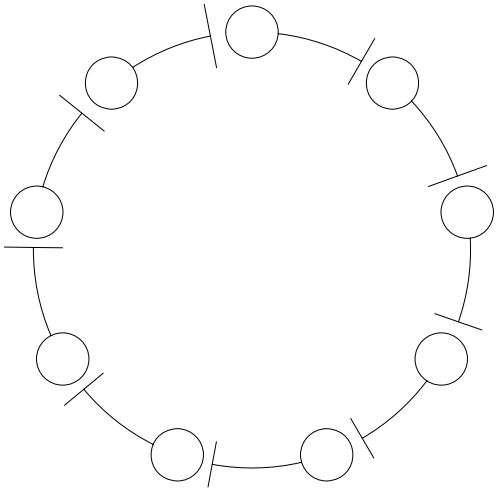
- **definitions**

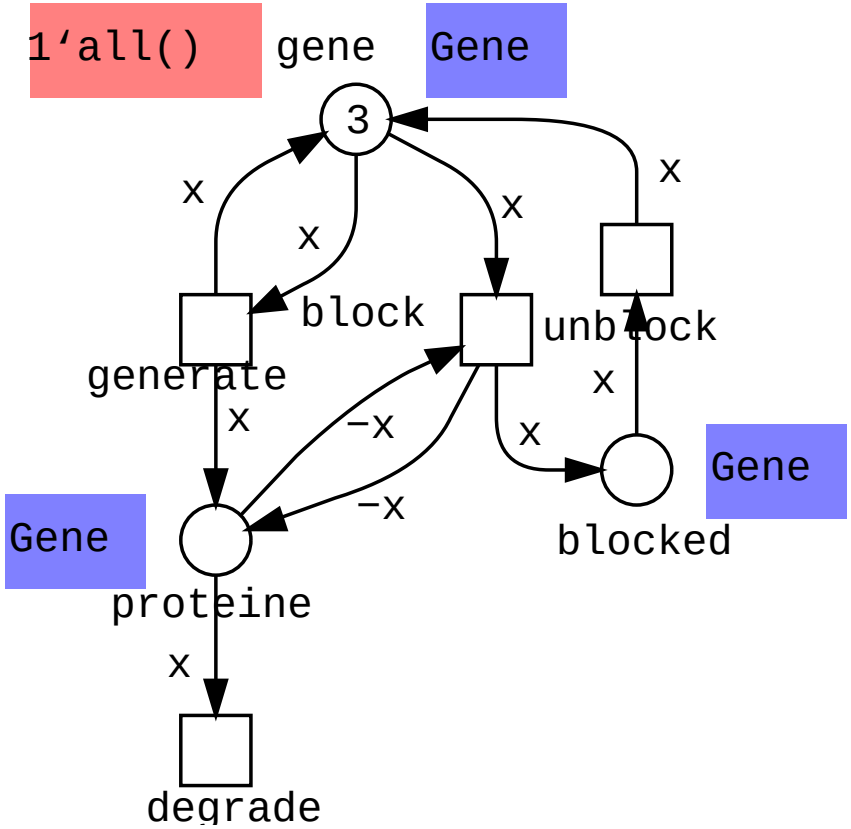
  colorset *Gene = enum a-c;*

  var *x : Gene;*

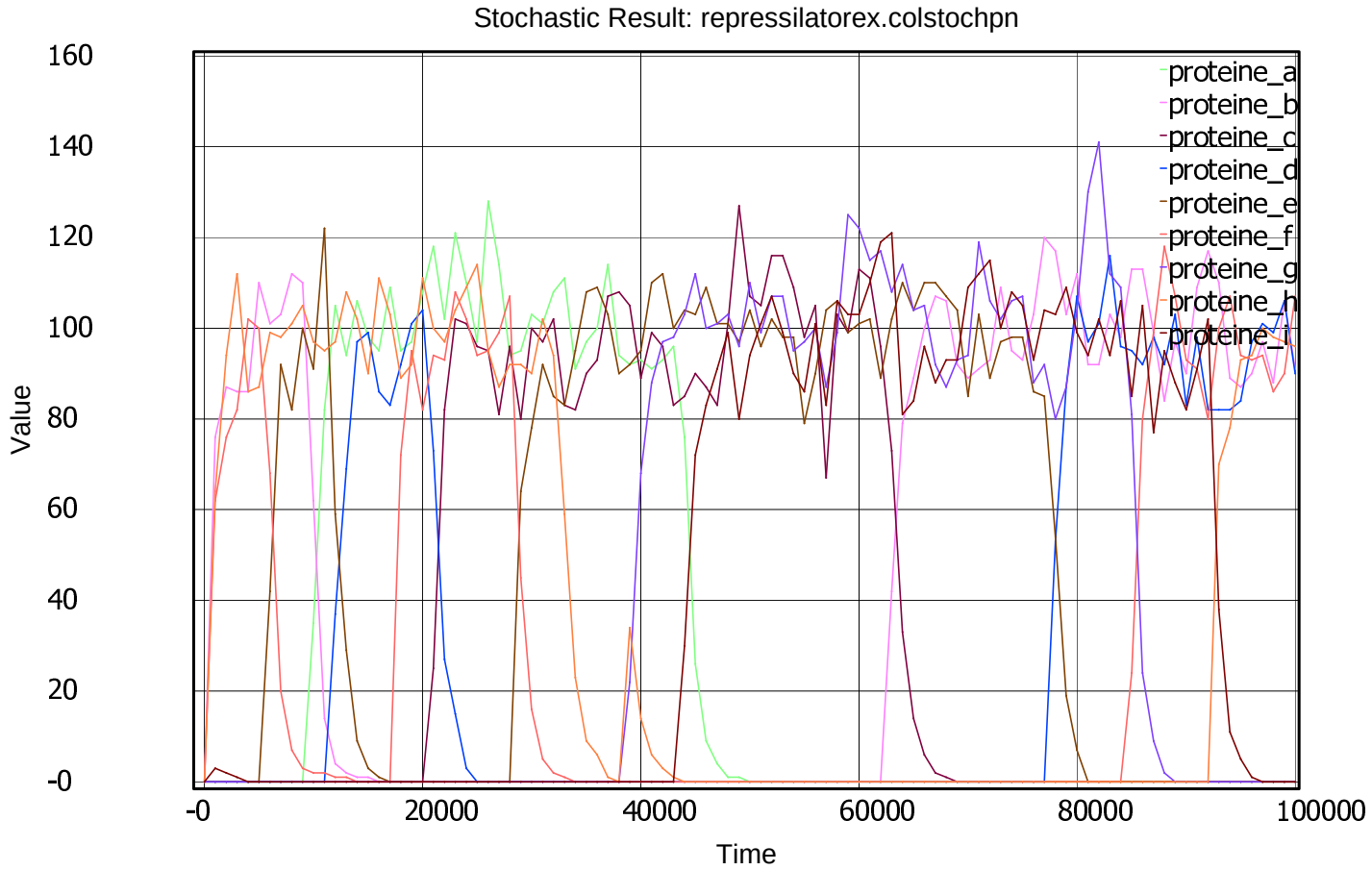
- **model scaling**

  colorset *Gene = enum a-i;*





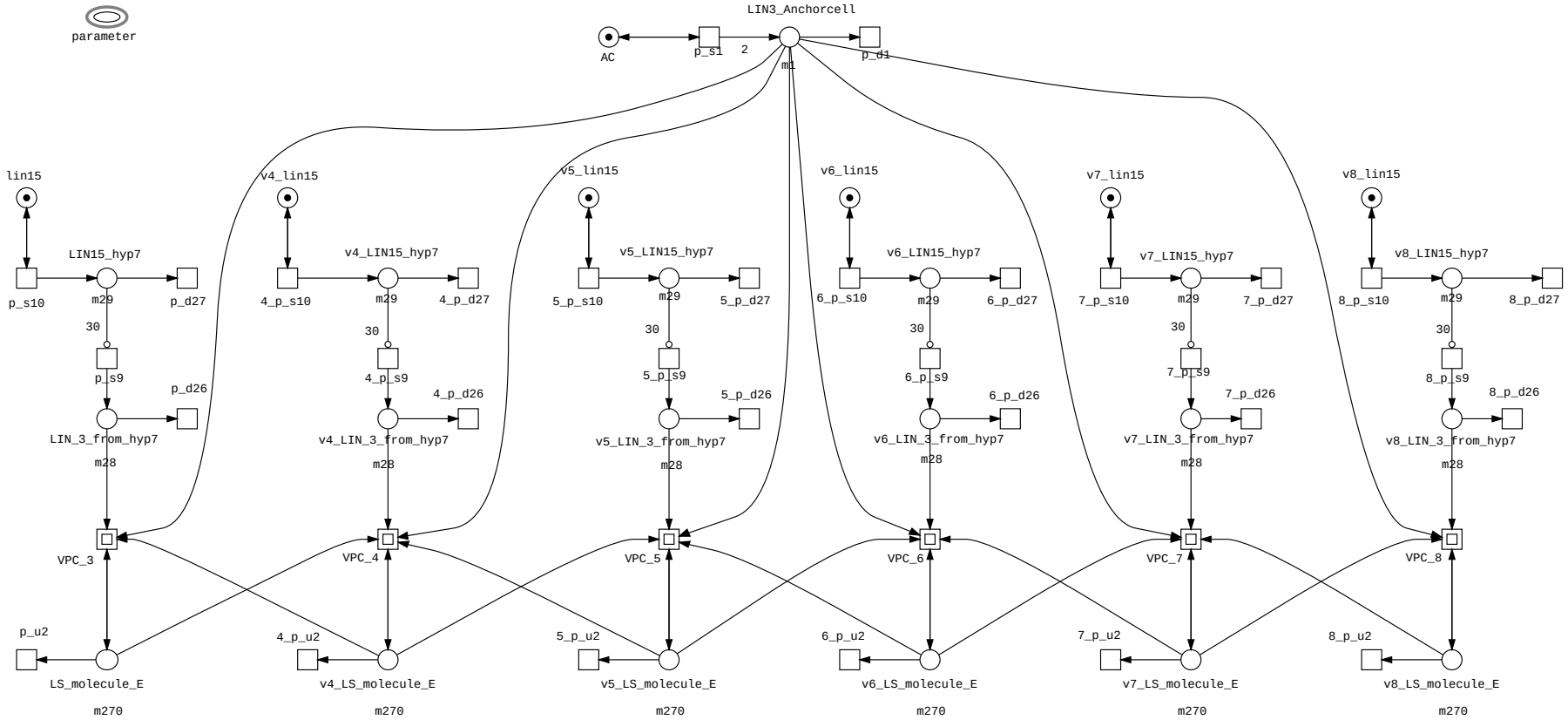
EX: REPRESSILATOR



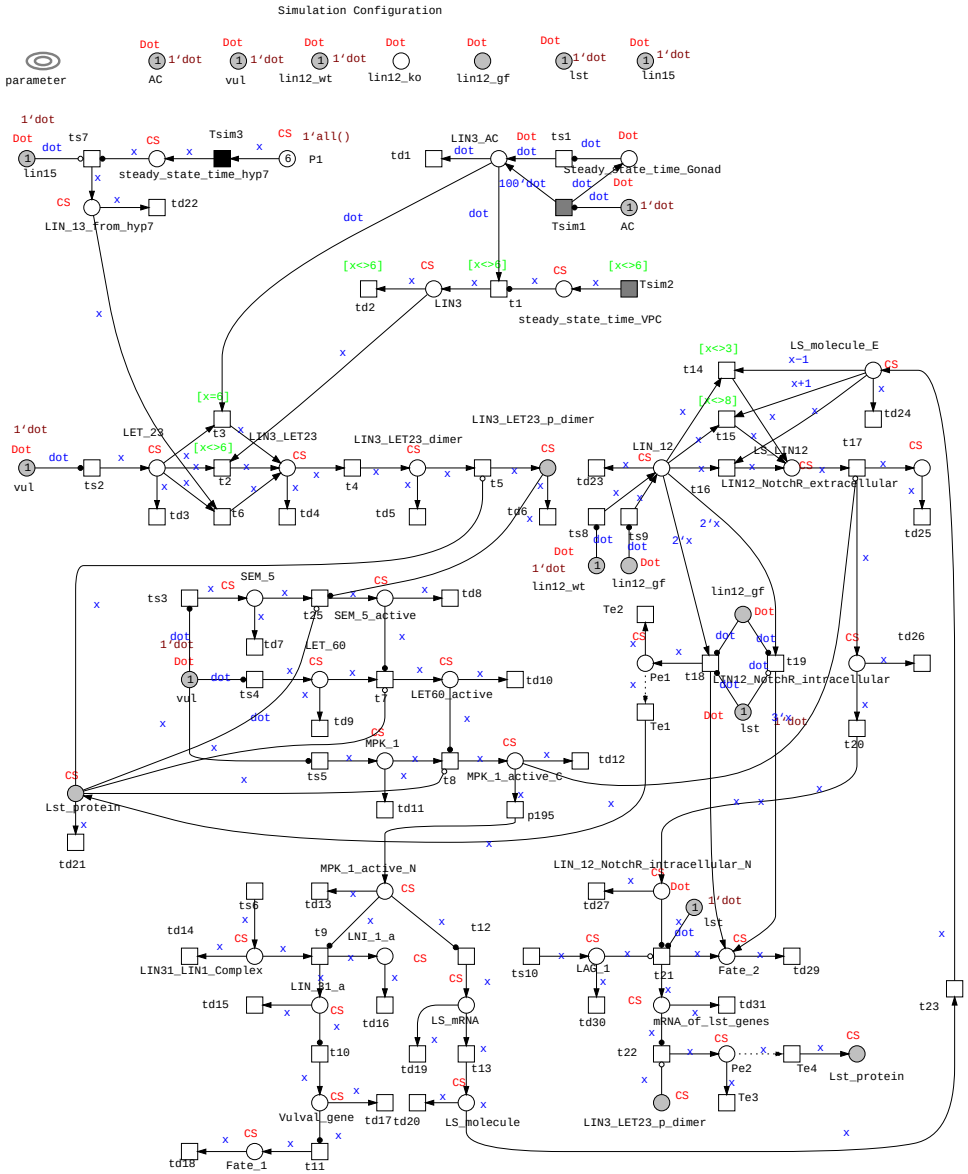
EX: C. ELEGANS

[LI ET AL. 2009]
[BONZANNI ET AL. 2009]

PLACES: 206
TRANSITIONS: 366

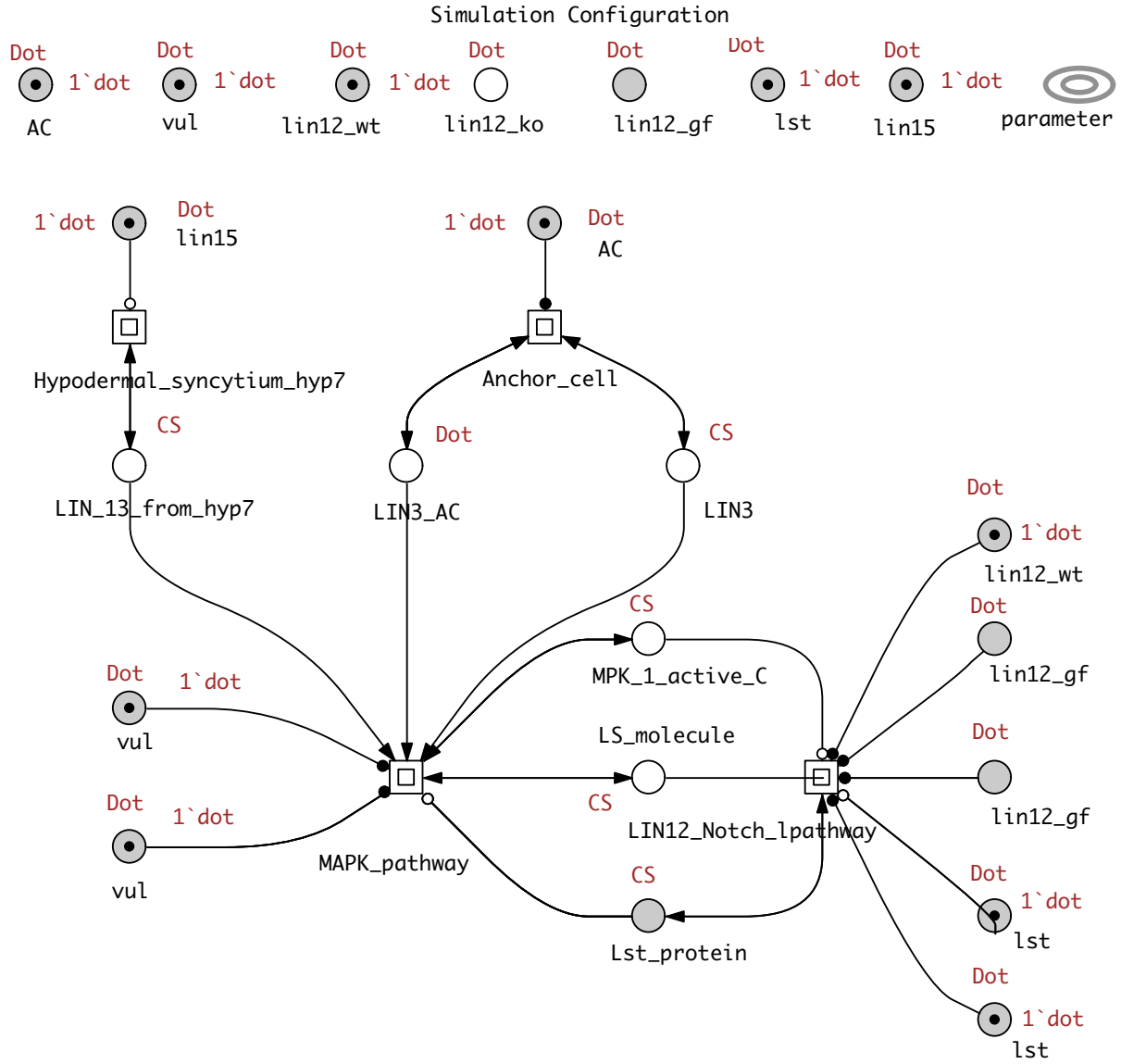


EX: C. ELEGANS, COLOURED



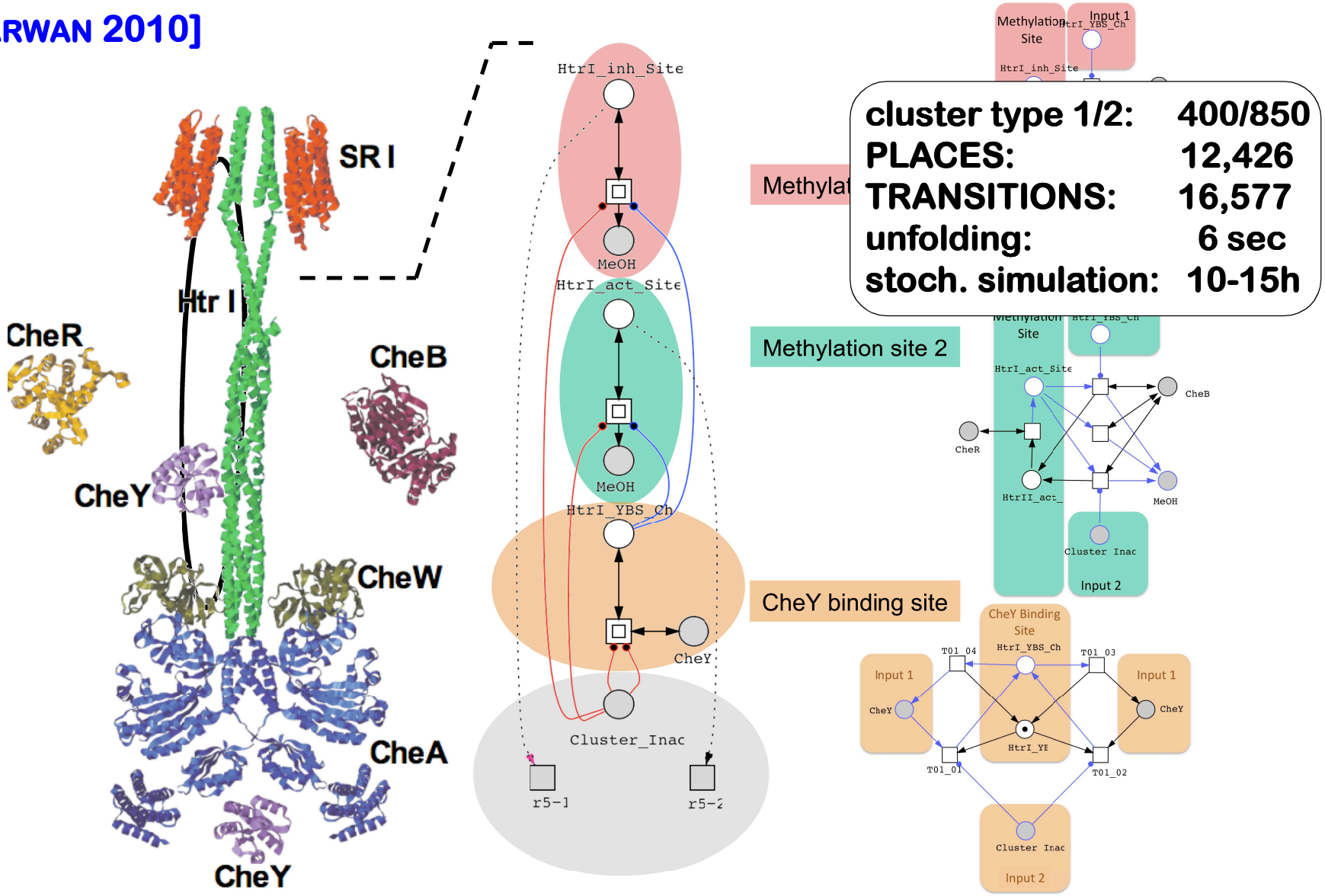
PLACES:	**44**
TRANSITIONS:	**72**

EX: C. ELEGANS, COLOURED & HIERARCHIES



EX - HALOBACTERIUM SALINARUM

[MARWAN 2010]



COLOURING A NET, BASIC STEPS

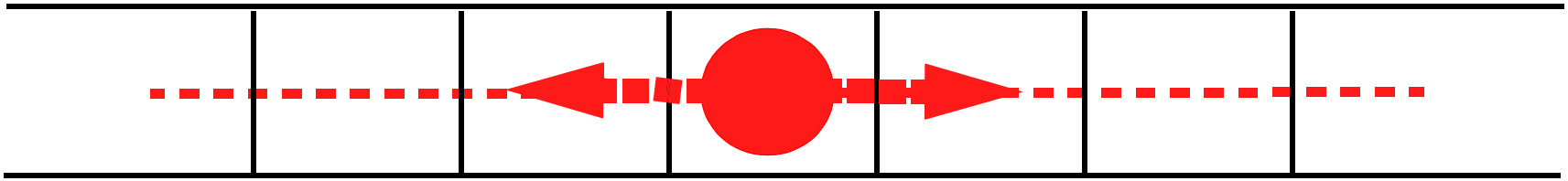
- **define**
  - -> *constants*
  - -> *colour sets*
  - -> *variables*
  - -> *functions*
- **assign**
  - -> *colour sets* *to* *places* *(default: colorset Dot)*
  - -> *expressions* *to* *arcs* *(default: dot, the only value of Dot)*
  - -> *guards* *to* *transitions* *(default: trivial guard „true”)*
- **set initial marking**
- **carefully test net behaviour**
- **. . . enjoy . . .**

EXAMPLE 1:

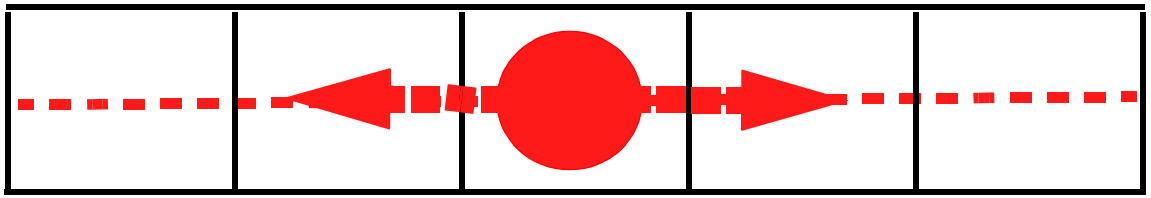
DIFFUSION IN SPACE

Richmond, 13/09/2011

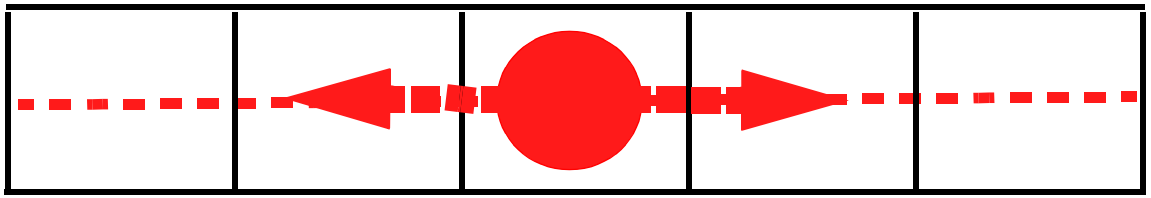
EX1: DIFFUSION - 1D

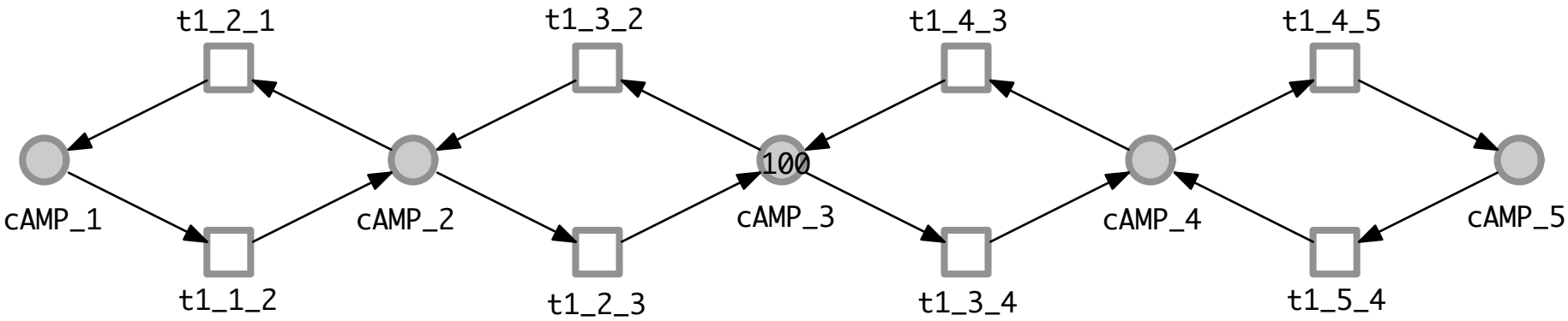


EX1: DIFFUSION - 1D



EX1: DIFFUSION - 1D





EX1: DIFFUSION - 1D

- **definitions**

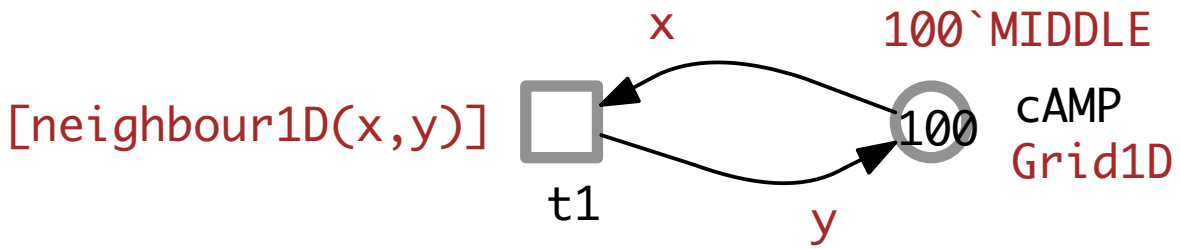
```
const D1 = 5;                // grid size
const MIDDLE = D1/2;
colorset Grid1D = 1-D1; // grid positions
var x,y : Grid1D;
```

- **definitions**

```
const D1 = 5;                // grid size
const MIDDLE = D1/2;
colorset Grid1D = 1-D1; // grid positions
var x,y : Grid1D;

function neighbour1D (Grid1D x,a) bool:
  // a is neighbour of x
  ( a=x-1 | a=x+1)  & (1<=a) & (a<=D1);
```

- **definitions**

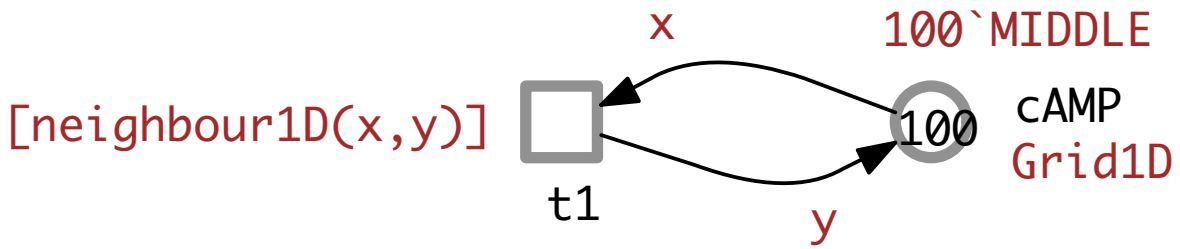
```
const D1 = 5;                // grid size
const MIDDLE = D1/2;
colorset Grid1D = 1-D1; // grid positions
var x,y : Grid1D;

function neighbour1D (Grid1D x,a) bool:
  // a is neighbour of x
  ( a=x-1 | a=x+1)  & (1<=a) & (a<=D1);
```



EX1: DIFFUSION - 1D

- **definitions**

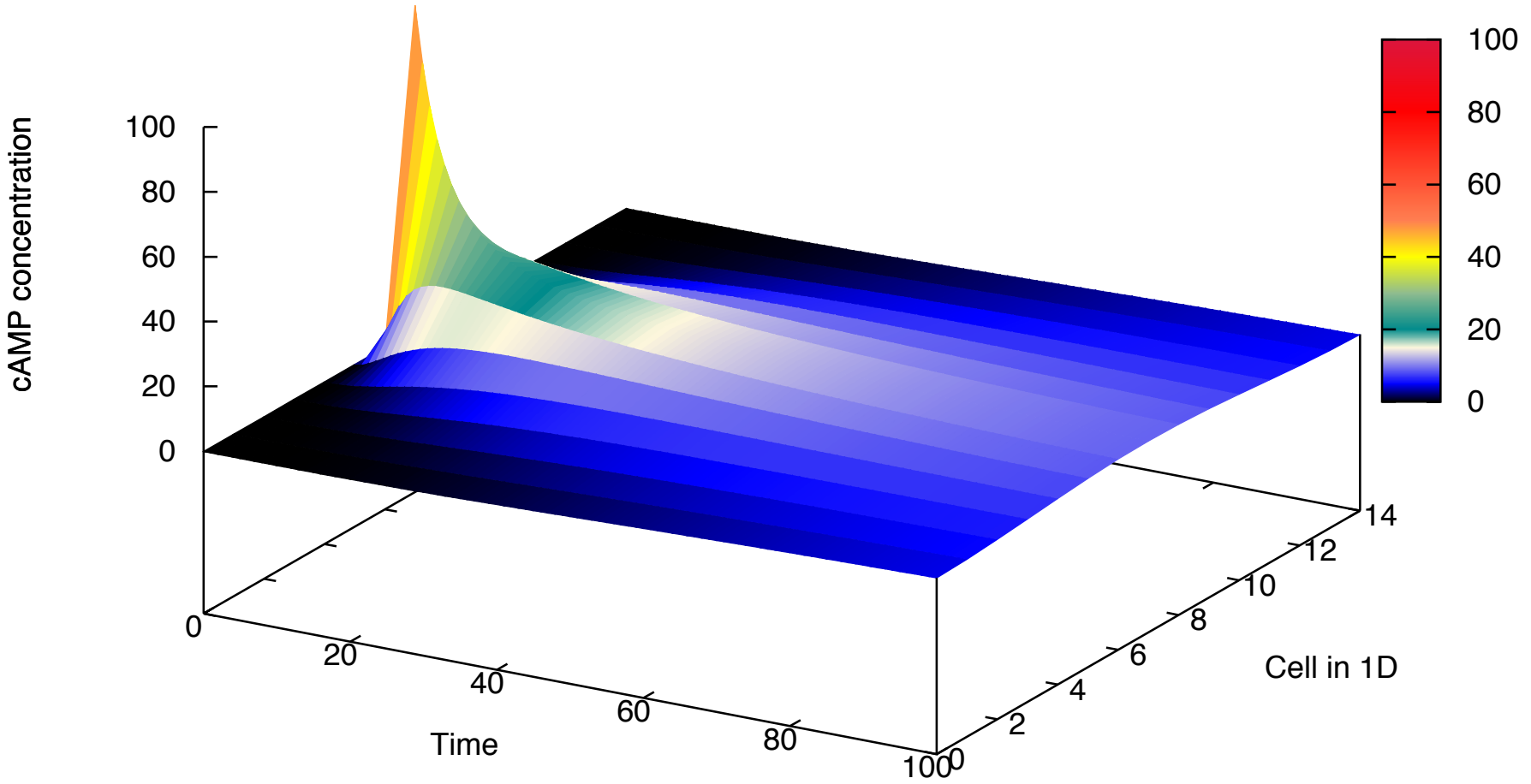
```
const D1 = 5;                // grid size
const MIDDLE = D1/2;
colorset Grid1D = 1-D1; // grid positions
var x,y : Grid1D;

function neighbour1D (Grid1D x,a) bool:
  // a is neighbour of x
  ( a=x-1 | a=x+1)  & (1<=a) & (a<=D1);
```



- **movement = changing colour**

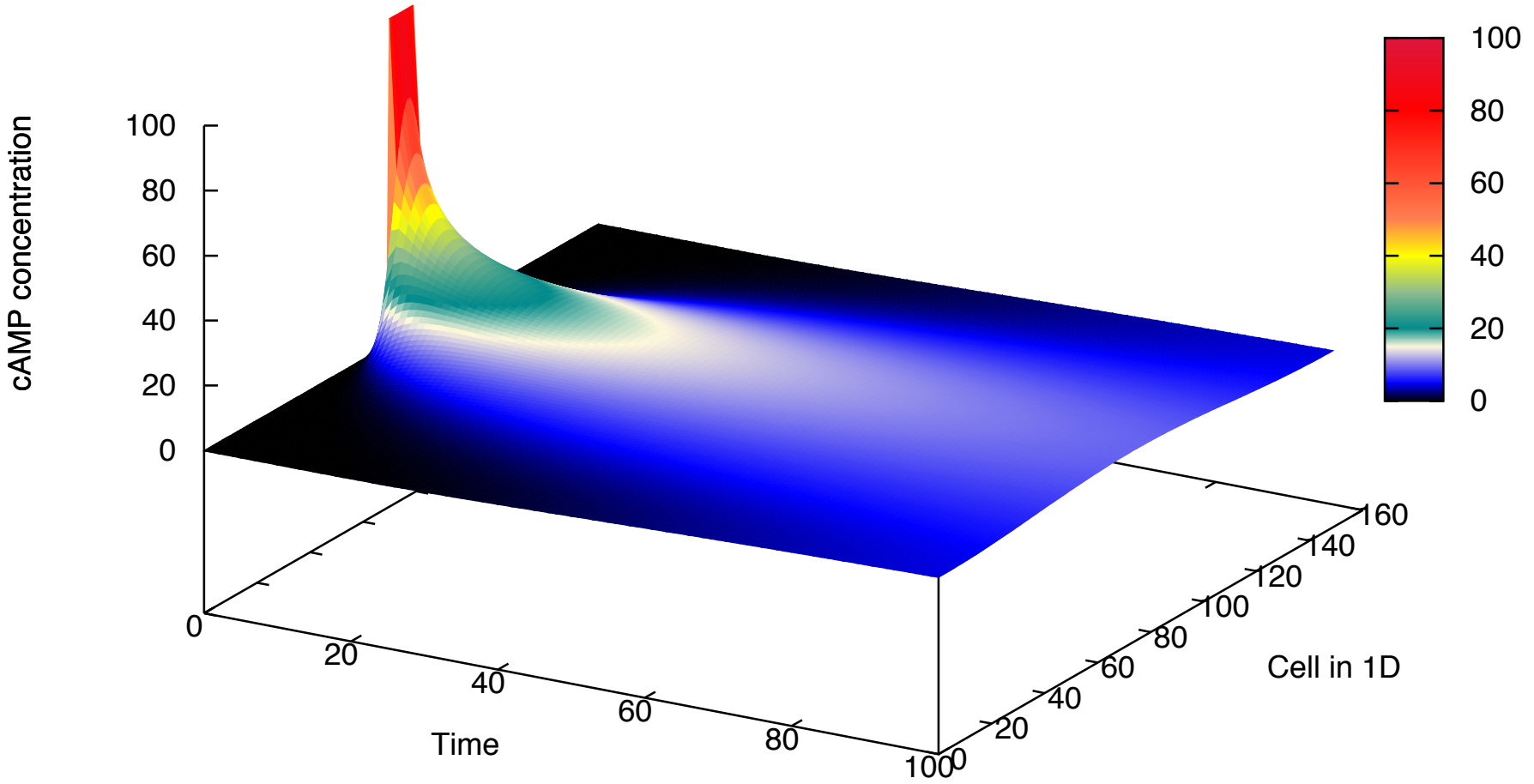
$$
\begin{aligned}
\frac{dc_1}{dt} &= k \cdot c_2 - k \cdot c_1 \\
\frac{dc_2}{dt} &= k \cdot c_1 + k \cdot c_3 - 2 \cdot k \cdot c_2 \\
\frac{dc_3}{dt} &= k \cdot c_2 + k \cdot c_4 - 2 \cdot k \cdot c_3 \\
\frac{dc_4}{dt} &= k \cdot c_3 + k \cdot c_5 - 2 \cdot k \cdot c_4 \\
\frac{dc_5}{dt} &= k \cdot c_4 - k \cdot c_5
\end{aligned}
$$

EX1: DIFFUSION - 1D



15 GRID POSITIONS

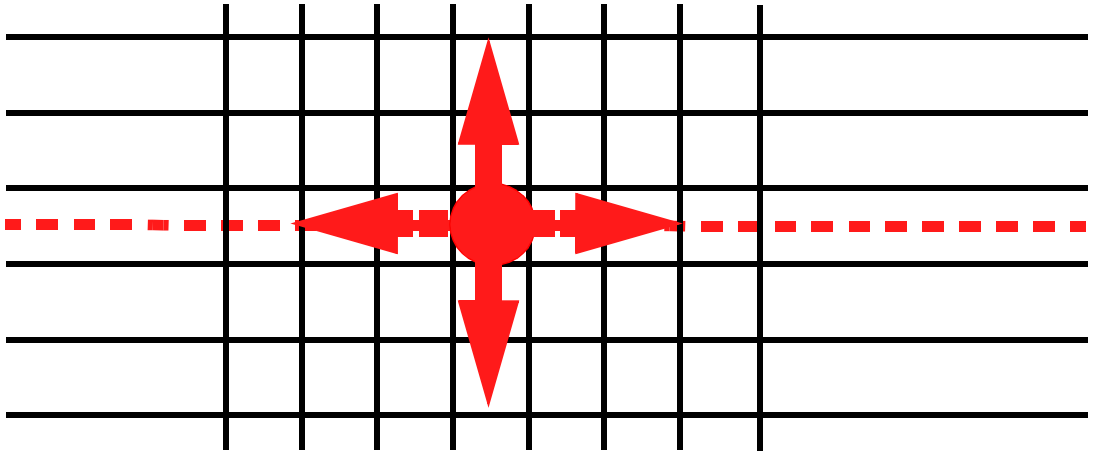
EX1: DIFFUSION - 1D



150 GRID POSITIONS, SCALING OF INITIAL MARKING AND RATES

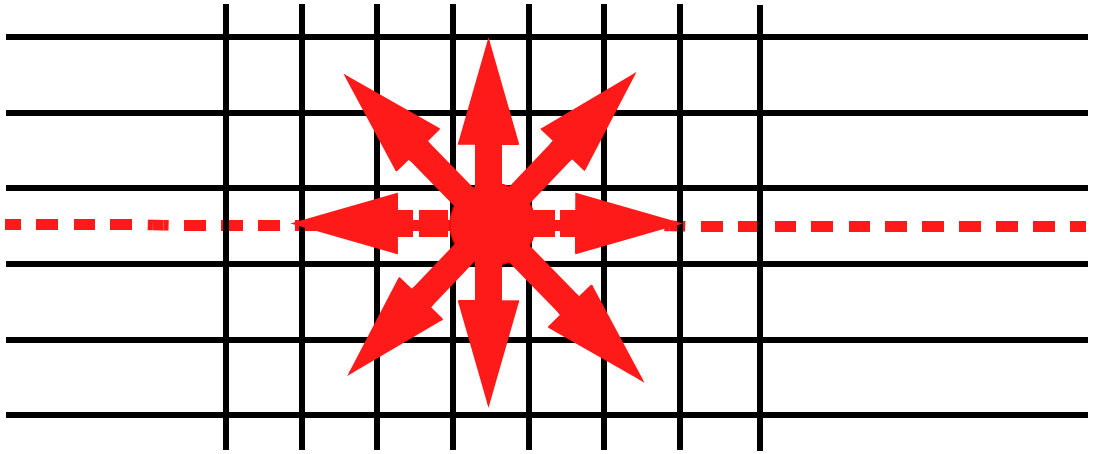
EX1: DIFFUSION - 2D

SCHEME

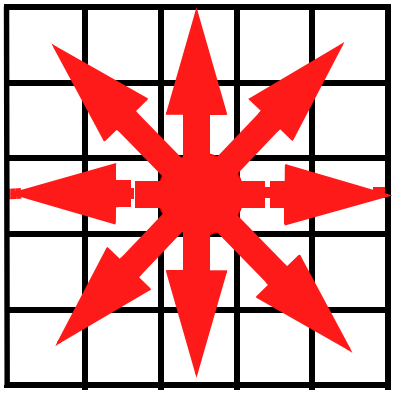


EX1: DIFFUSION - 2D

SCHEME

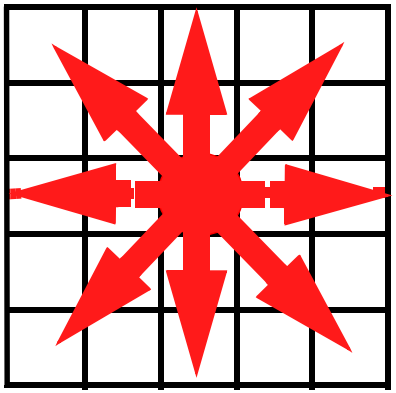


SCHEME



EX1: DIFFUSION - 2D

SCHEME

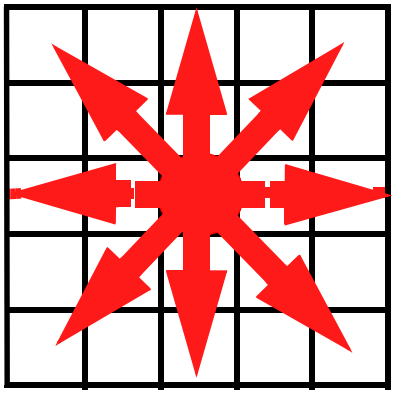


definitions

const *D1 = 5;* *// grid size first dimension*

const *D2 = D1;* *// grid size second dimension*

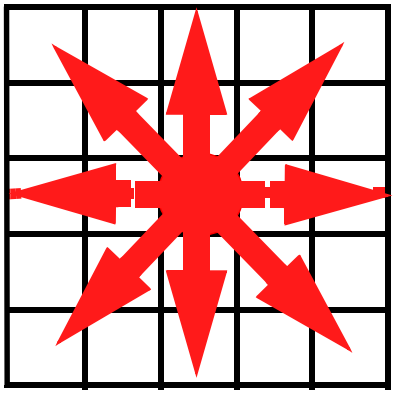
SCHEME



definitions

```
const D1 = 5;                    // grid size first dimension
const D2 = D1;                   // grid size second dimension
const MIDDLE = D1/2;

colorset CD1 = 1-D1;             // row index
colorset CD2 = 1-D2;             // column index
colorset Grid2D = CD1 x CD2;     // 2D grid
```

SCHEME



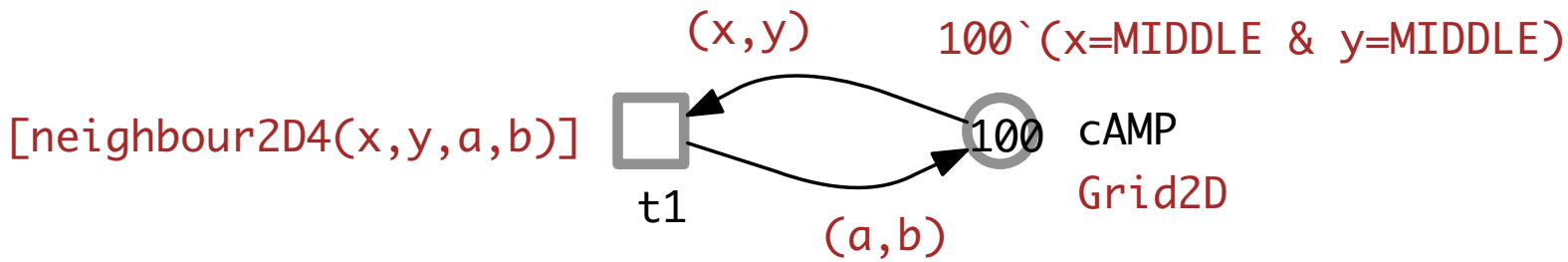
definitions

```
const D1 = 5;                    // grid size first dimension
const D2 = D1;                   // grid size second dimension
const MIDDLE = D1/2;

colorset CD1 = 1-D1;             // row index
colorset CD2 = 1-D2;             // column index
colorset Grid2D = CD1 x CD2;     // 2D grid

var x, a : CD1;
var y, b : CD2;
```

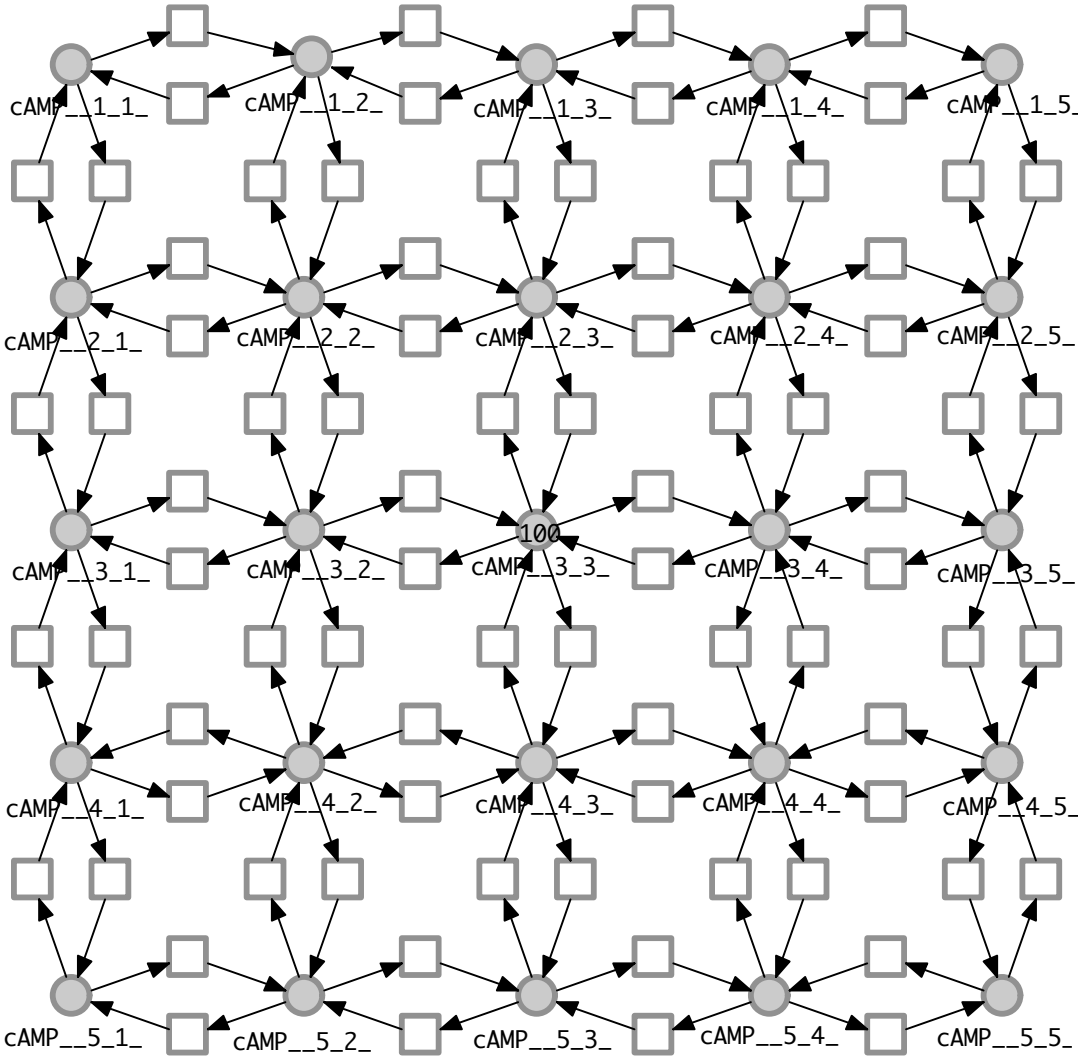
four neighbours

function *neighbour2D4 (CD1 x, CD2 y, CD1 a, CD2 b)* ***bool****:*

```
// (a,b) is one of the up to four neighbours of (x,y)
  ( (a=x & b=y-1) | (a=x & b=y+1)
  |  (b=y & a=x-1) | (b=y & a=x+1) )
 & (1<=a & a<=D1) & (1<=b & b<=D2);
```



EX1: DIFFUSION - 2D4 NEIGHBOURHOOD



EX1: DIFFUSION - 2D8 NEIGHBOURHOOD

eight neighbours

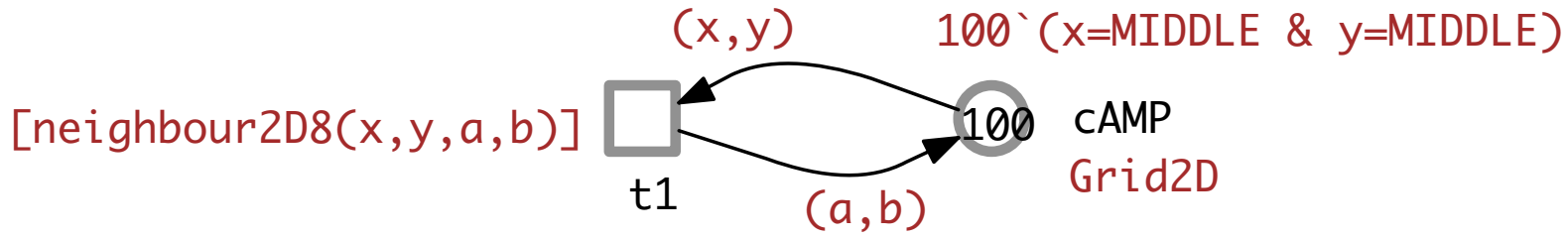
function *neighbour2D8 (CD1 x, CD2 y, CD1 a, CD2 b)* ***bool****:*

// (a,b) is one of the up to eight neighbours of (x,y)

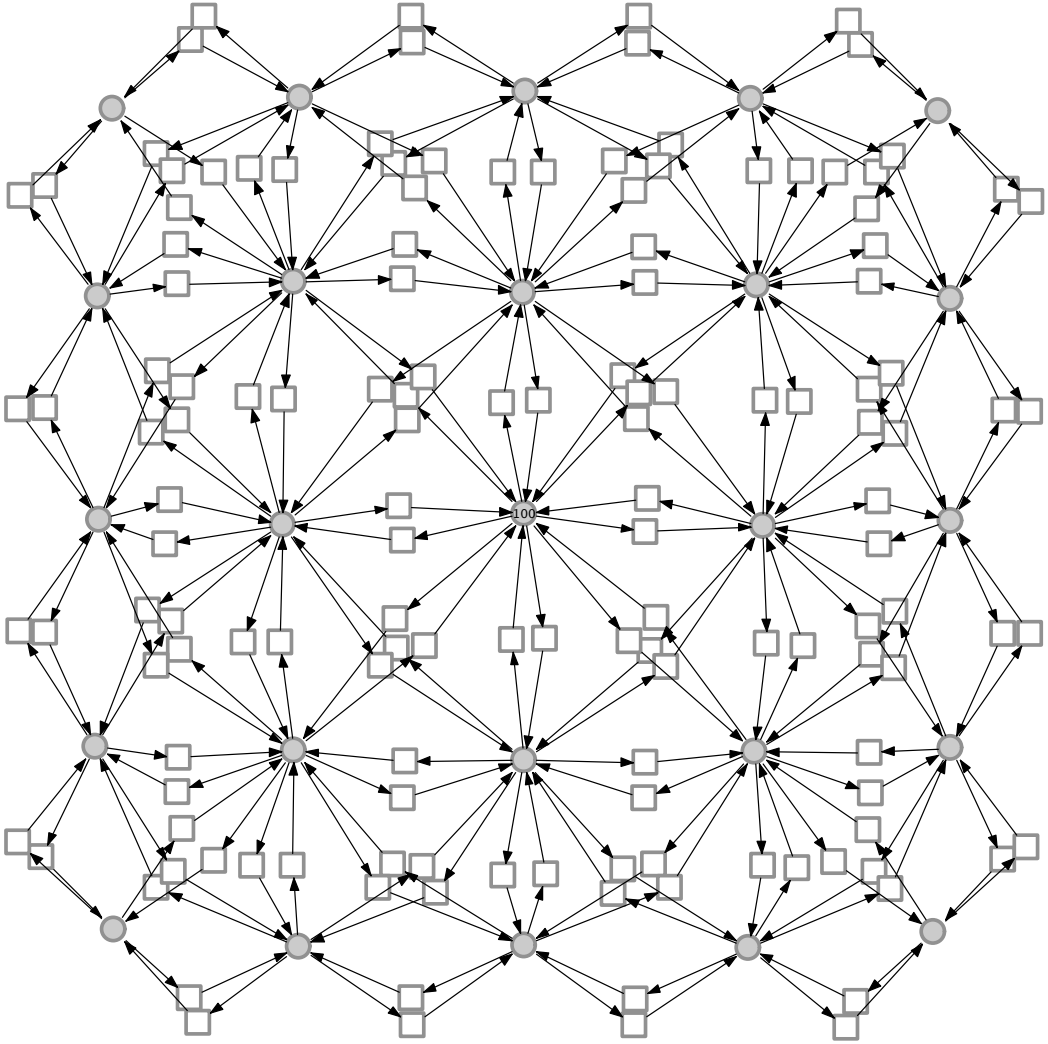
(a=x-1 | a=x | a=x+1) & (b = y-1 | b=y | b=y+1)

& (!(a=x & b=y))

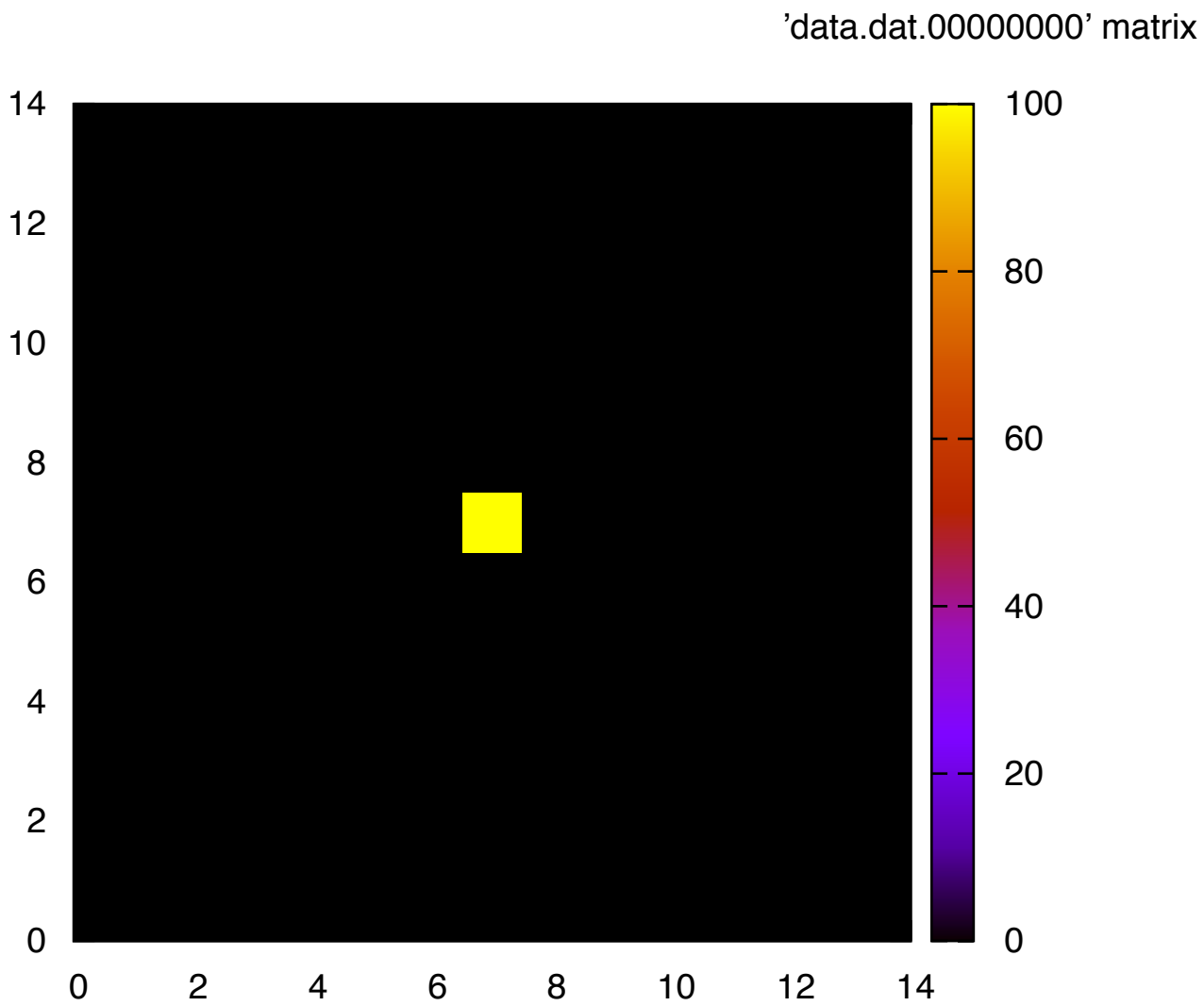
& (1<=a & a<=D1) & (1<=b & b<=D2);



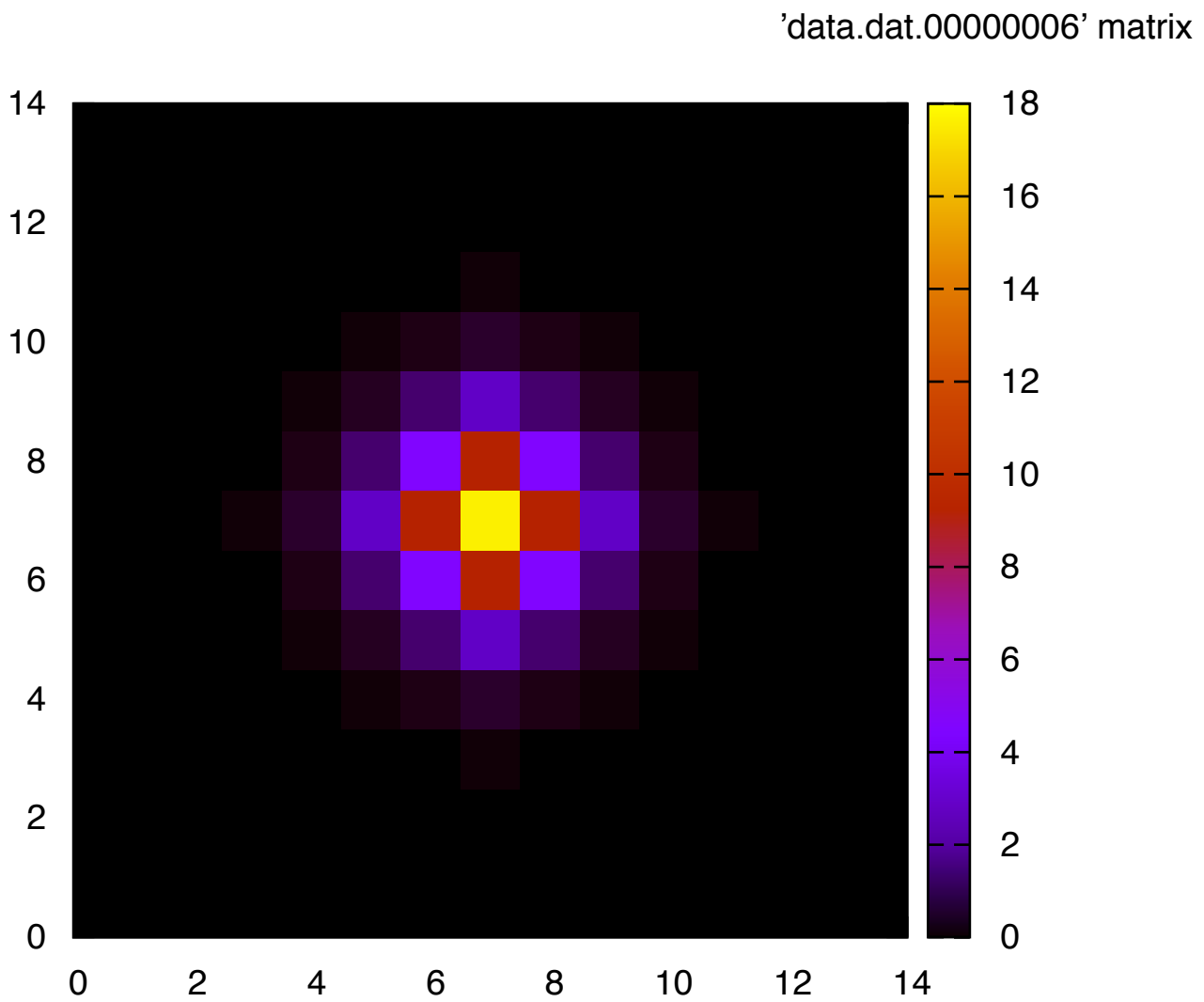
EX1: DIFFUSION - 2D8 NEIGHBOURHOOD



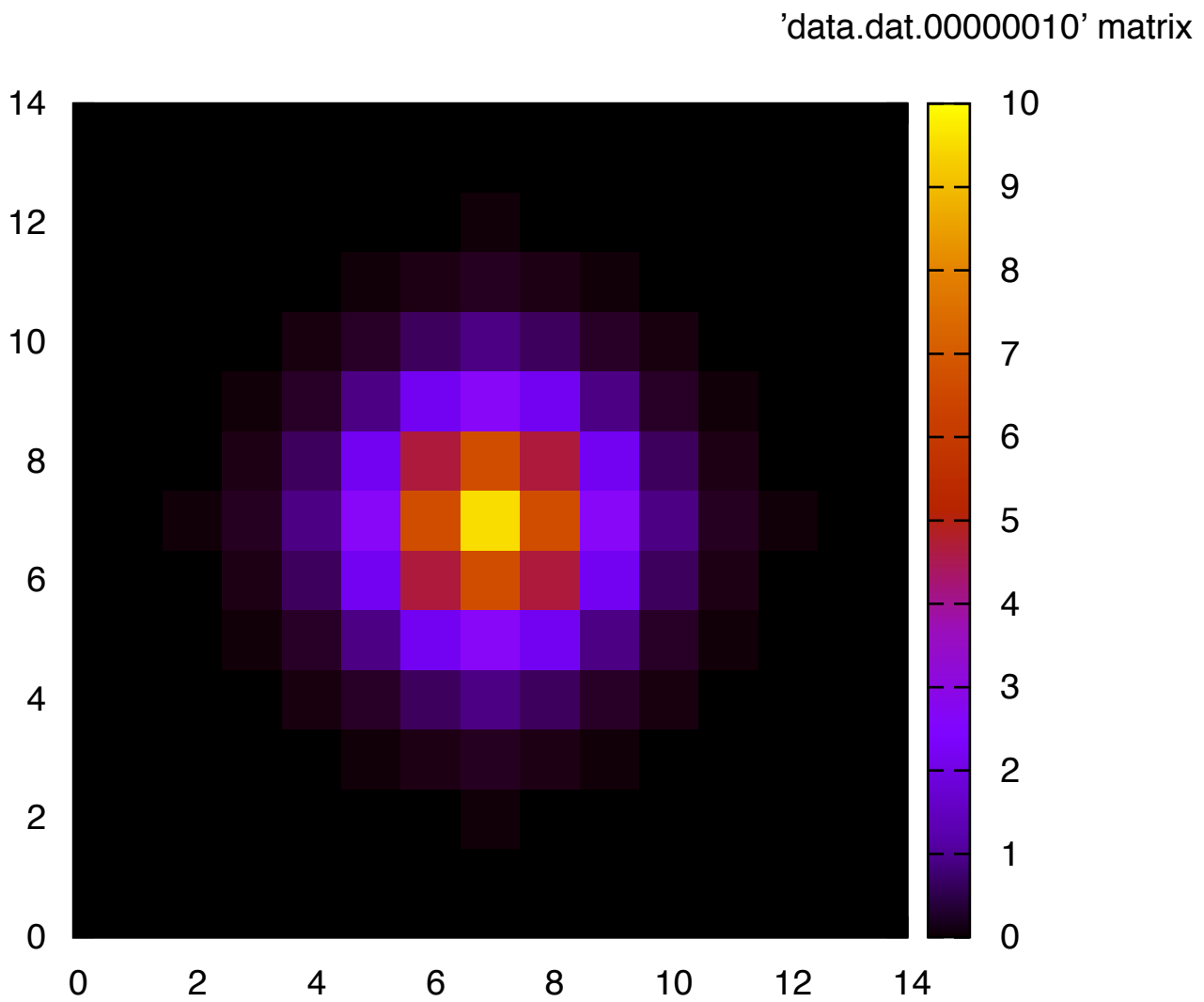
EX1: DIFFUSION - 2D4 NEIGHBOURHOOD, 15X15



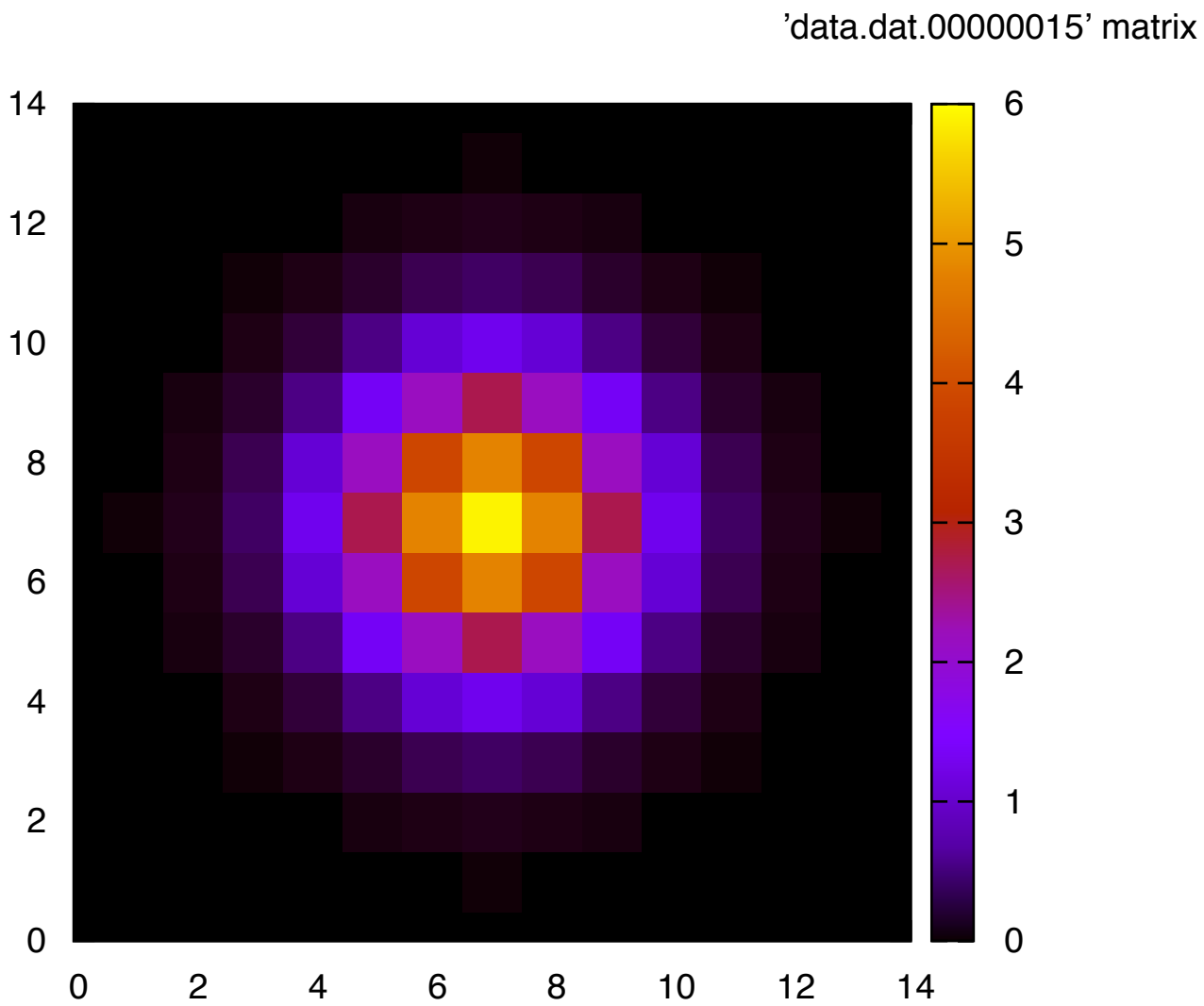
EX1: DIFFUSION - 2D4 NEIGHBOURHOOD, 15X15



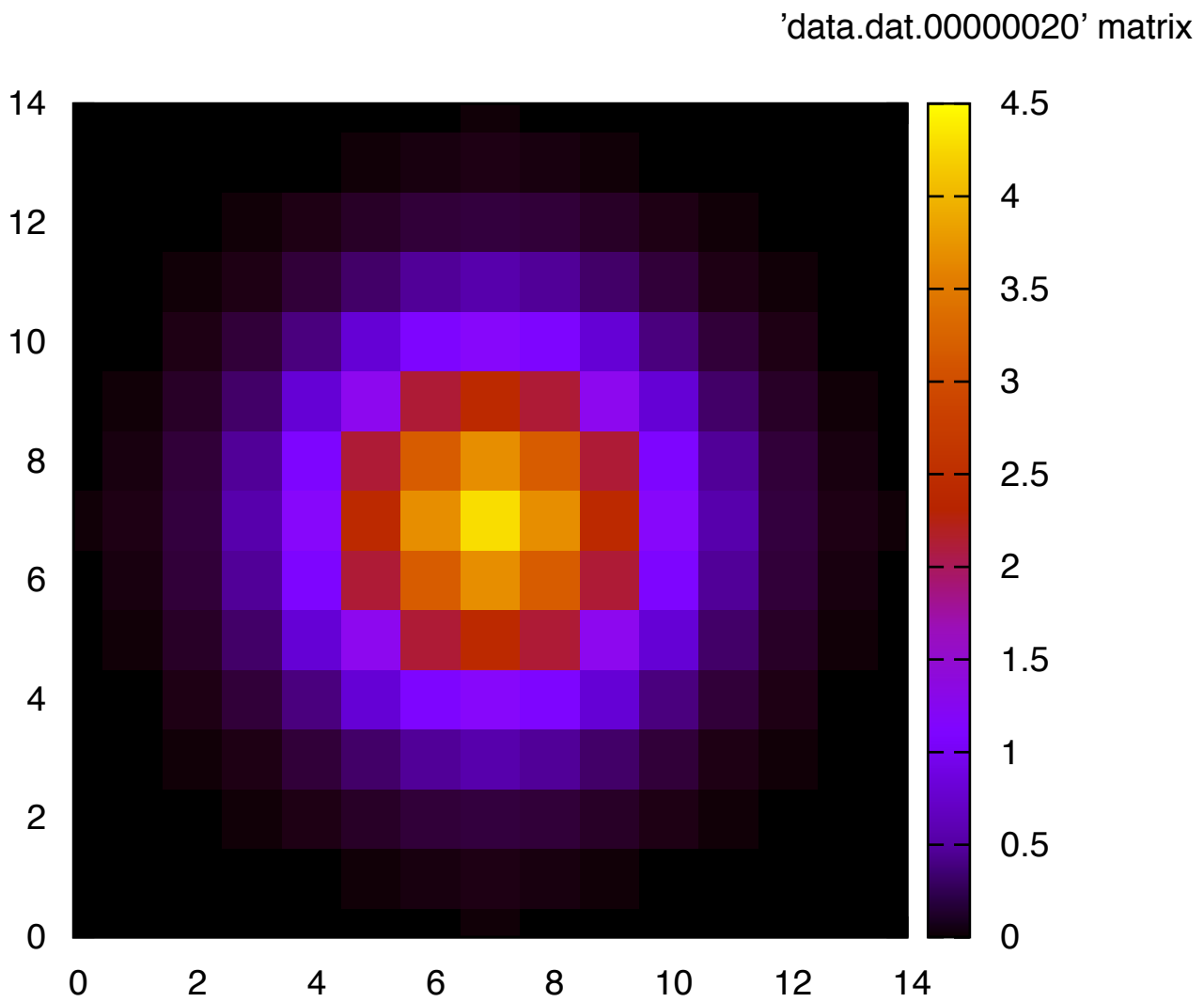
EX1: DIFFUSION - 2D4 NEIGHBOURHOOD, 15X15



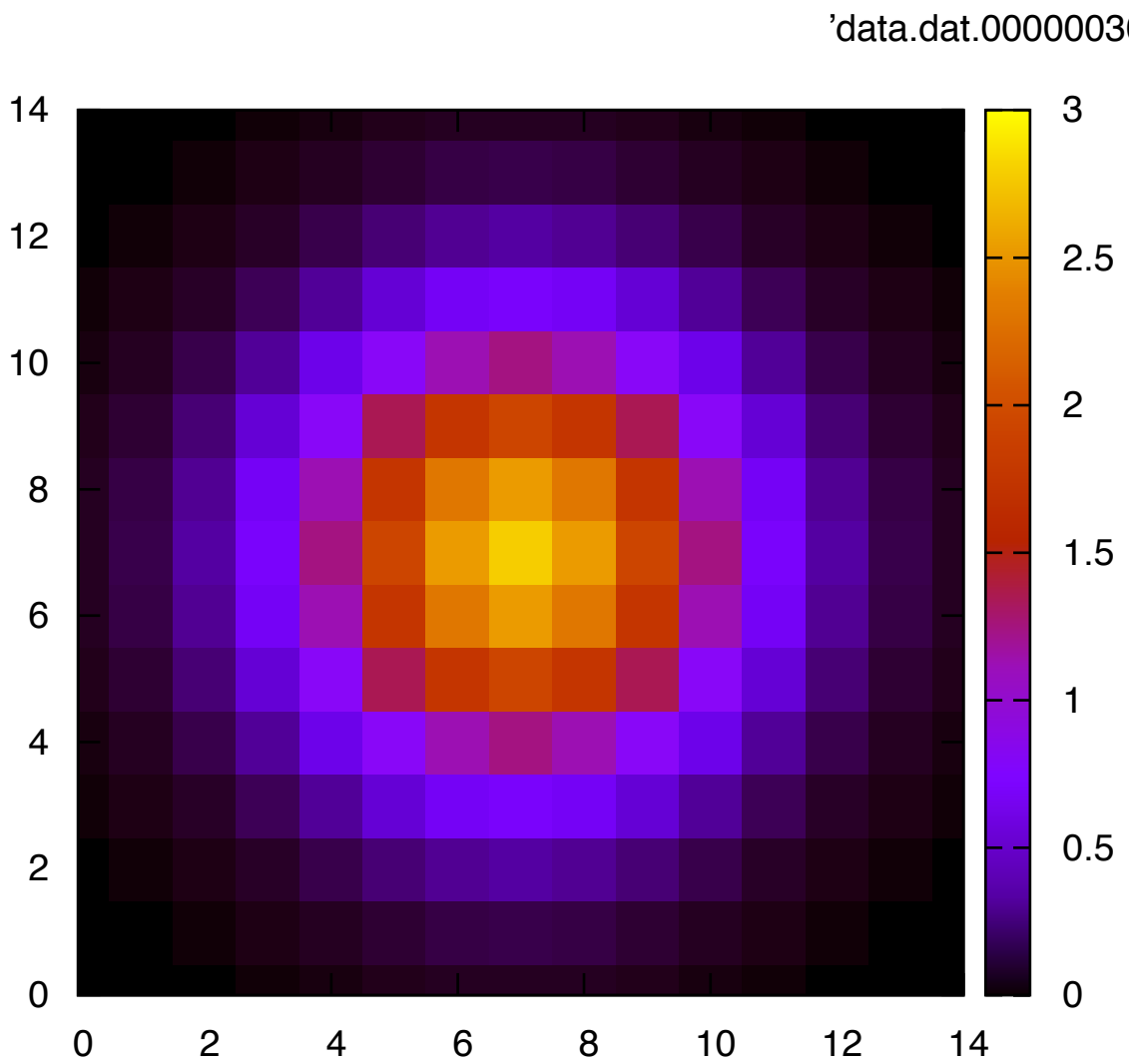
EX1: DIFFUSION - 2D4 NEIGHBOURHOOD, 15X15



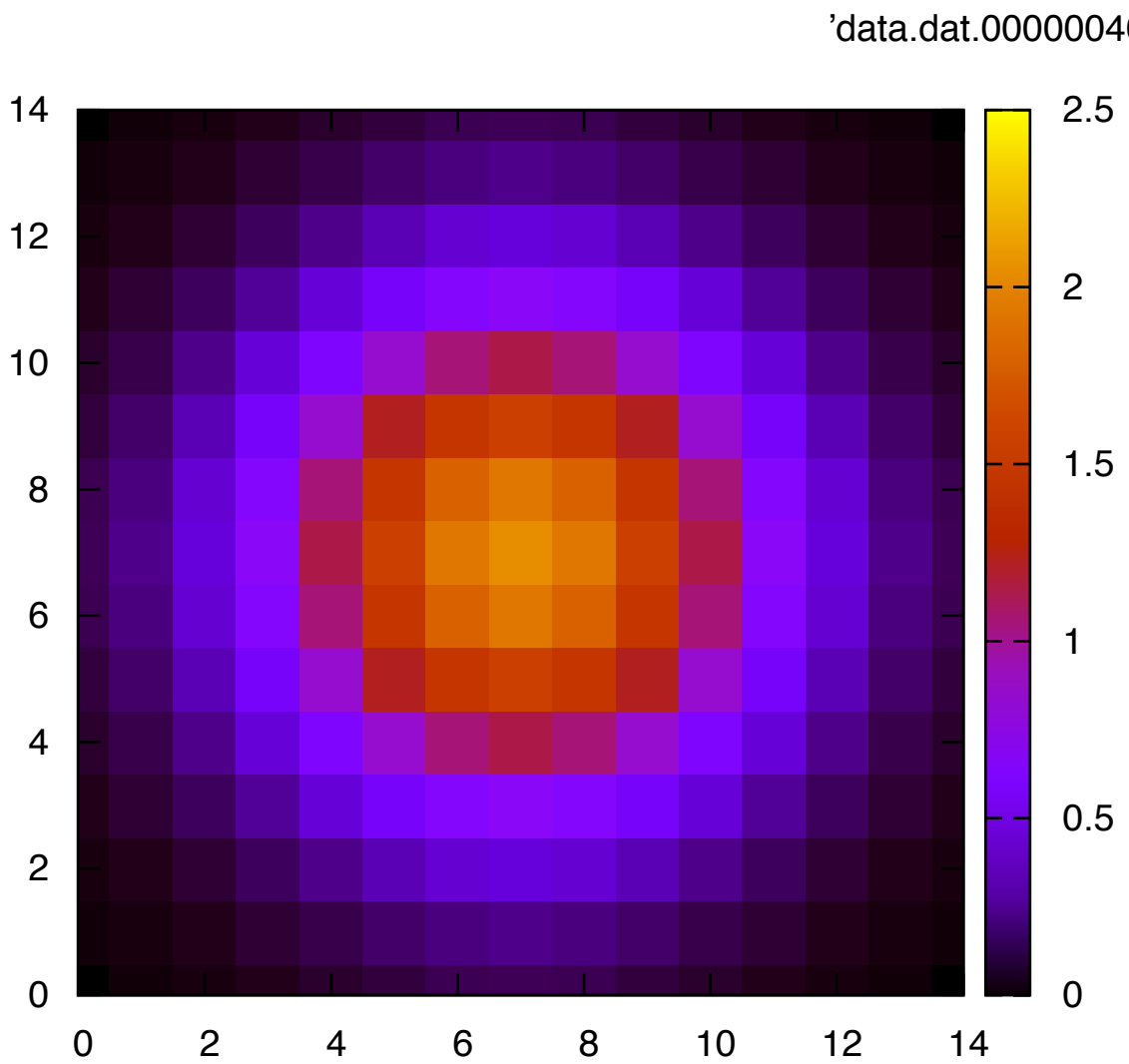
EX1: DIFFUSION - 2D4 NEIGHBOURHOOD, 15X15



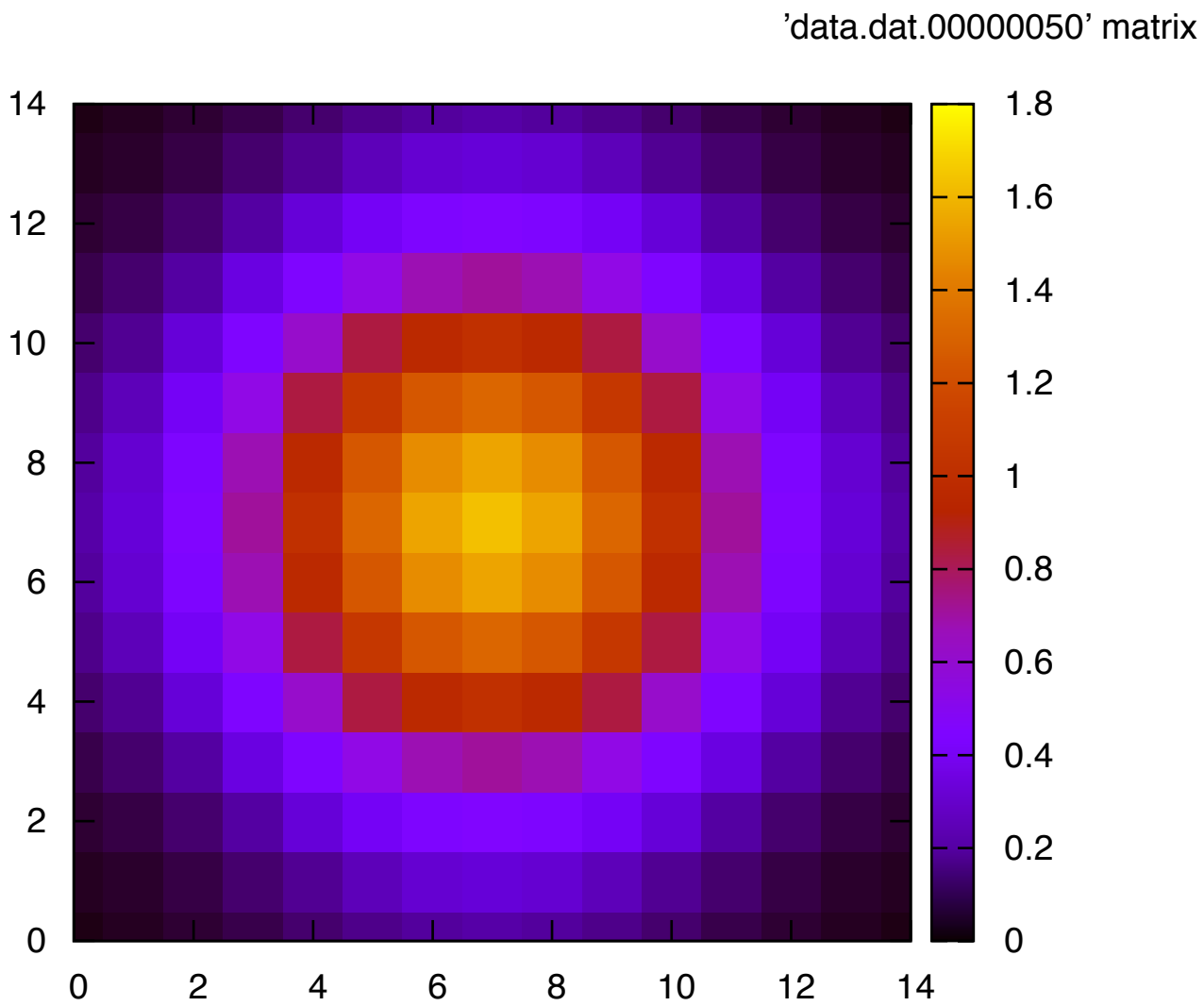
EX1: DIFFUSION - 2D4 NEIGHBOURHOOD, 15X15



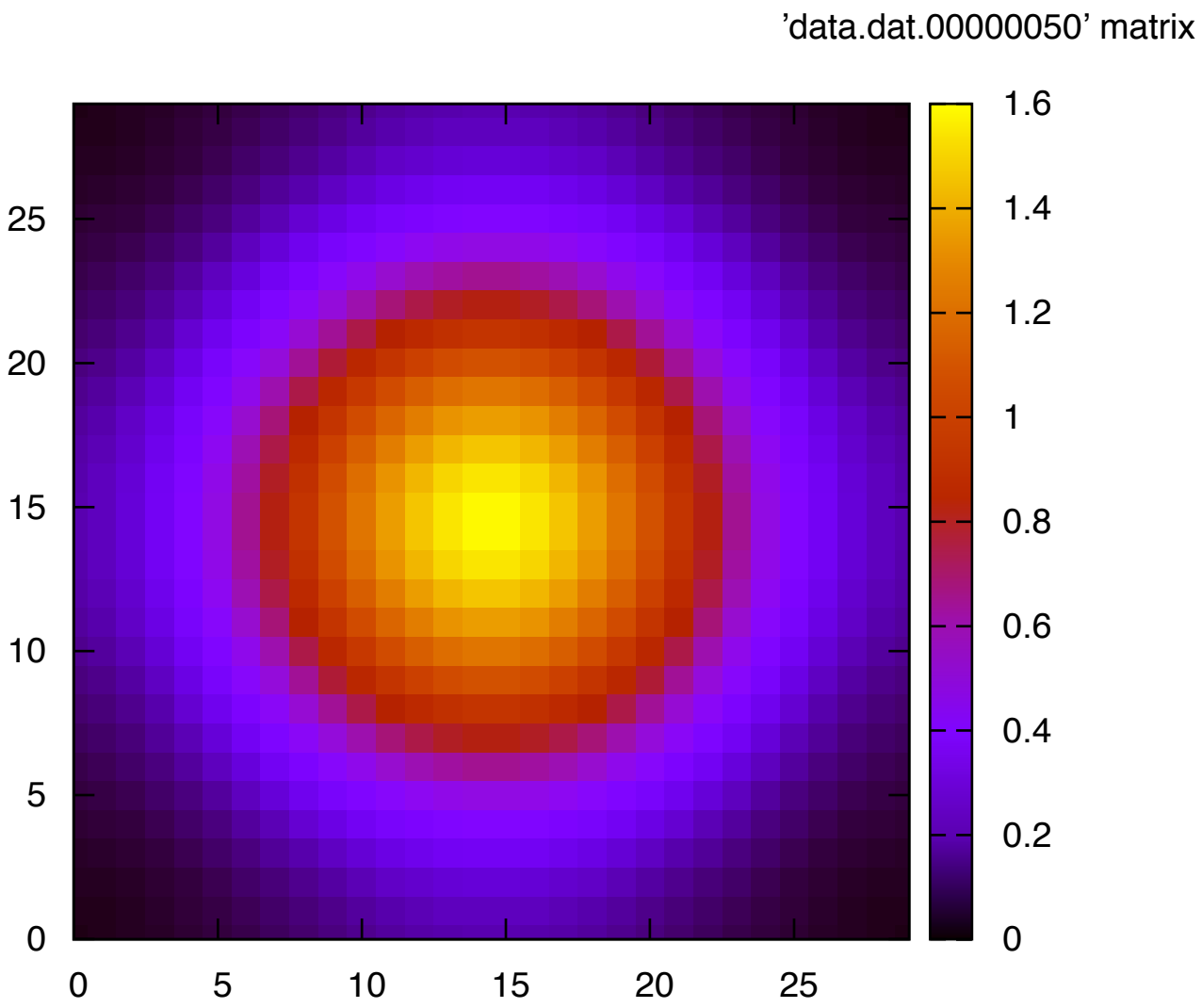
EX1: DIFFUSION - 2D4 NEIGHBOURHOOD, 15X15



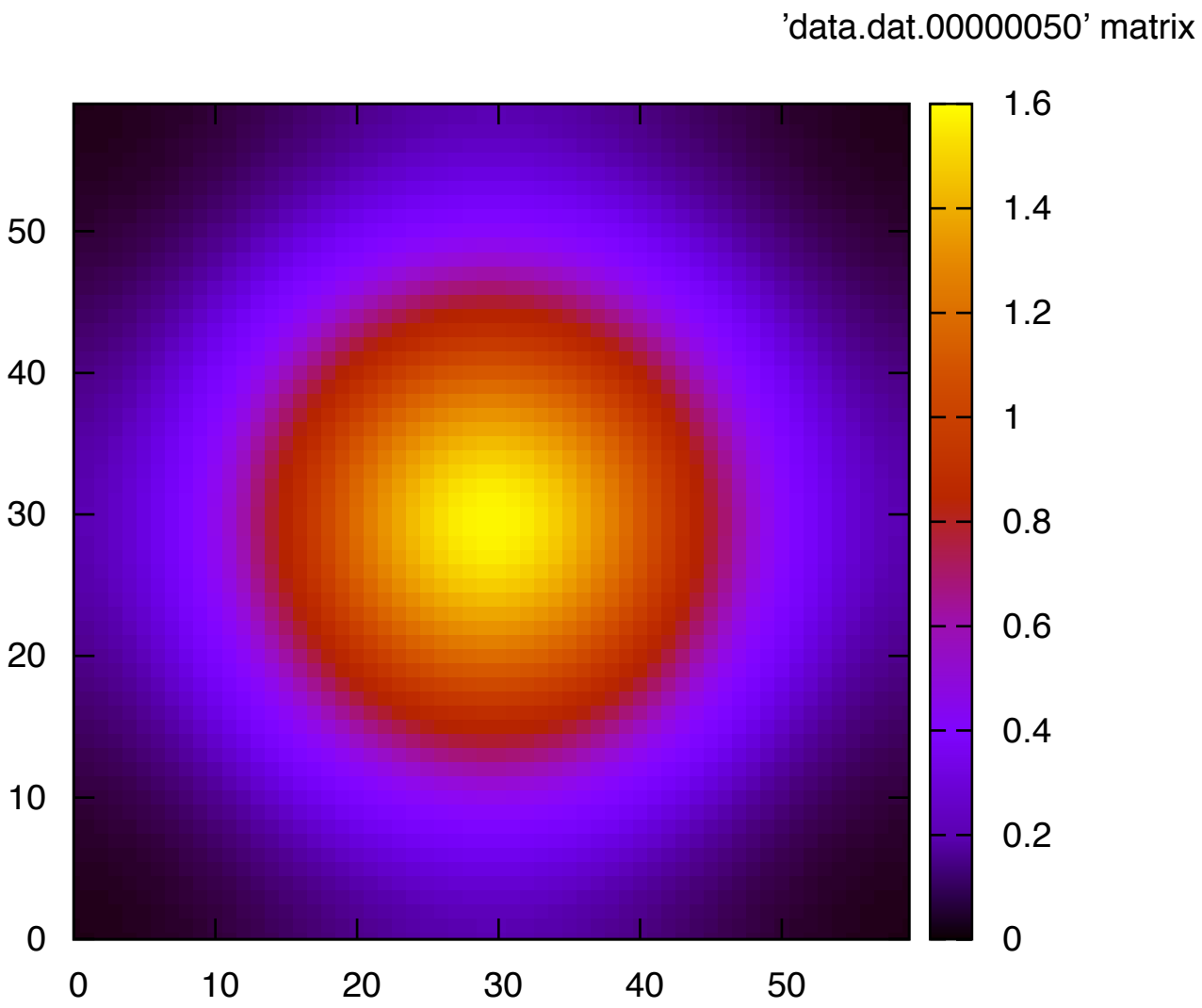
EX1: DIFFUSION - 2D4 NEIGHBOURHOOD, 15X15



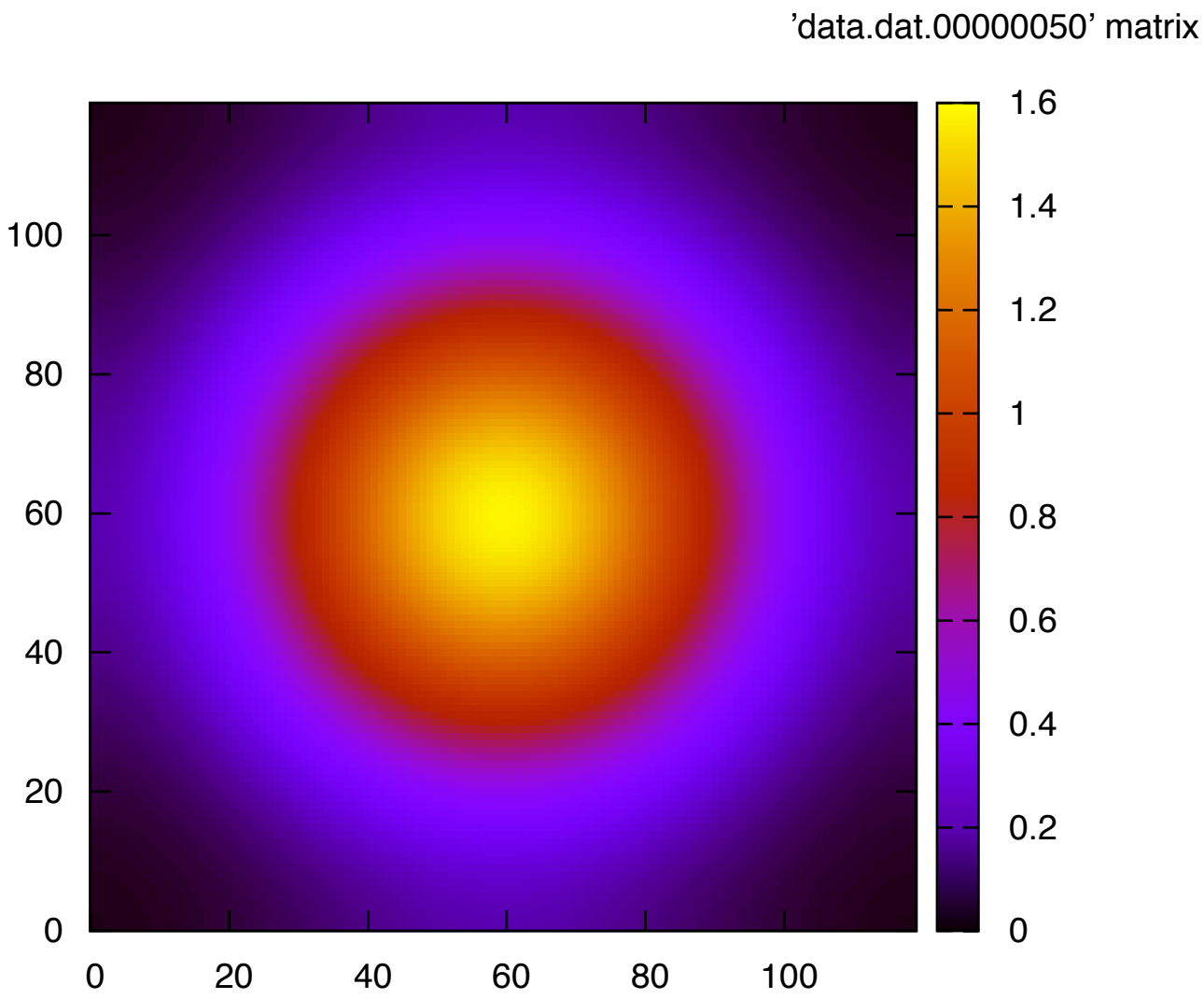
EX1: DIFFUSION - 2D4 NEIGHBOURHOOD, 30X30



EX1: DIFFUSION - 2D4 NEIGHBOURHOOD, 60X60



EX1: DIFFUSION - 2D4 NEIGHBOURHOOD, 120X120



EXAMPLE 2:

TURING PATTERNS

EX2: TURING PATTERNS

"How the Leopard Got Its Spots"

- **morphogenesis**
  - -> *developmental pattern formation in bio systems*
  - -> *the process that controls the organized spatial distribution of cells*
  - -> *tiger stripes, leopard spots, the precisely spaced rows of alligator teeth, etc.*

- **Turing's theory of biological pattern formation, 1952**
  - -> *patterns form as result of*

    the interactions between two chemicals
    that spread throughout a system at different rates

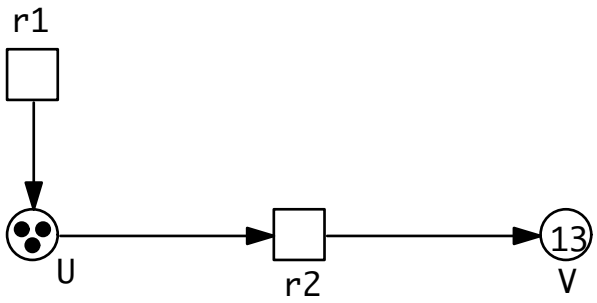
- **highly simplified and idealised take on biological patterning**

- **mathematical challenge**
  - -> *For which parameters do stable/oscillating Turing patterns exist ?*
  - -> *analysis of stability, multistability, bifurcation of non-linear PDE*

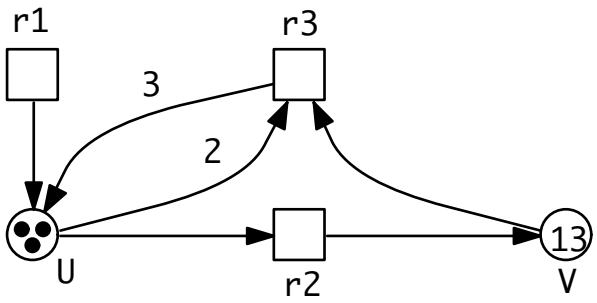
r1: -> U

r1: -> U

r2: U -> V

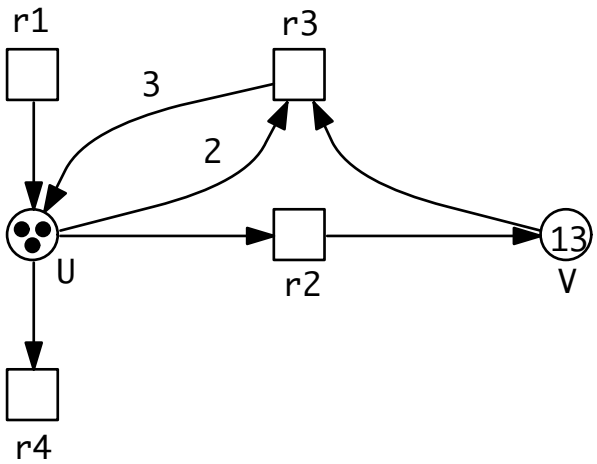


r1: -> U
r2: U -> V
r3: 2U + V -> 3U



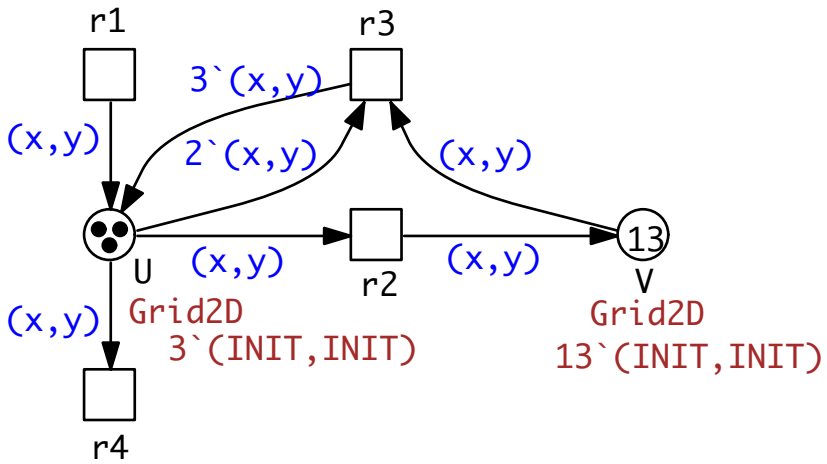
EX2: TURING PATTERNS

r1: -> U
r2: U -> V
r3: 2U + V -> 3U
r4: U ->



r1: -> U
r2: U -> V
r3: 2U + V -> 3U
r4: U ->

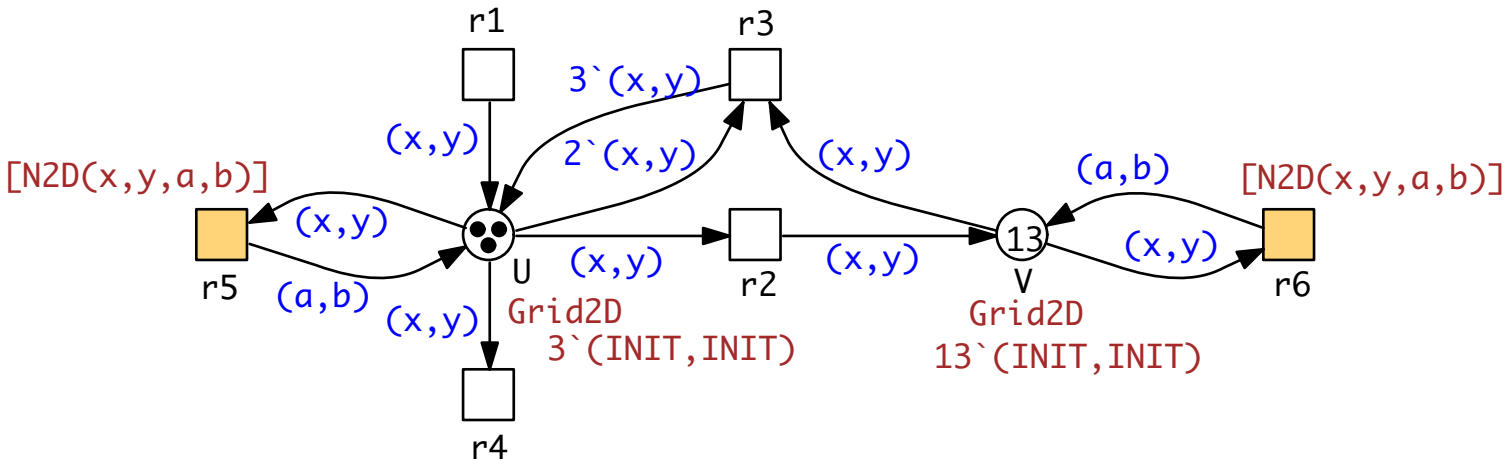
adding SPACE



EX2: TURING PATTERNS

r1: -> U
r2: U -> V
r3: 2U + V -> 3U
r4: U ->

diffusion:
r5: U_{xy} -- $1/h^2$ --> U_{ab}
r6: V_{xy} -- D/h^2 --> V_{ab}

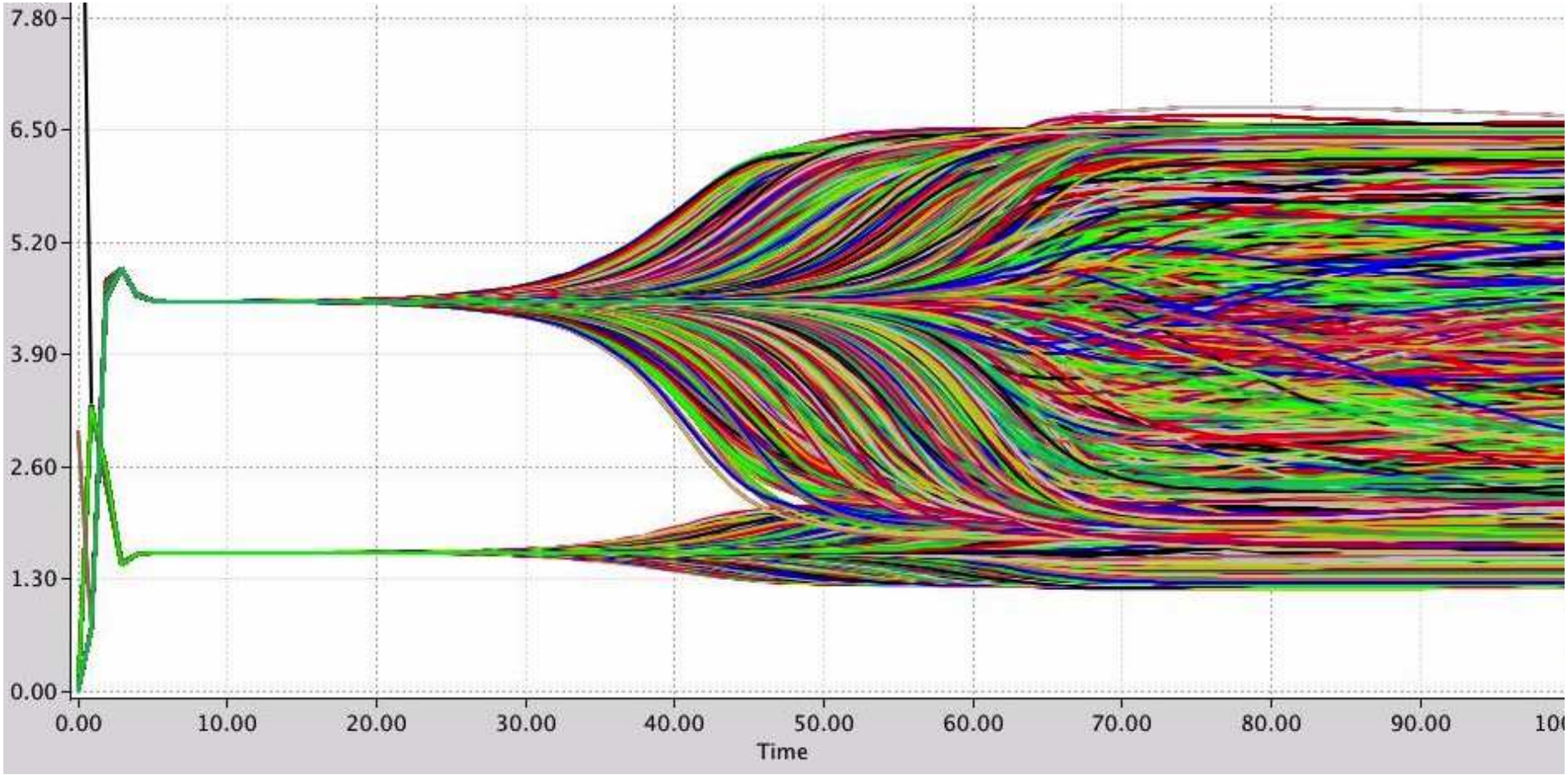


r1 - r4 follow mass action kinetics with rate constants:
r1: a, r2: b, r3: 1, r4: 1;

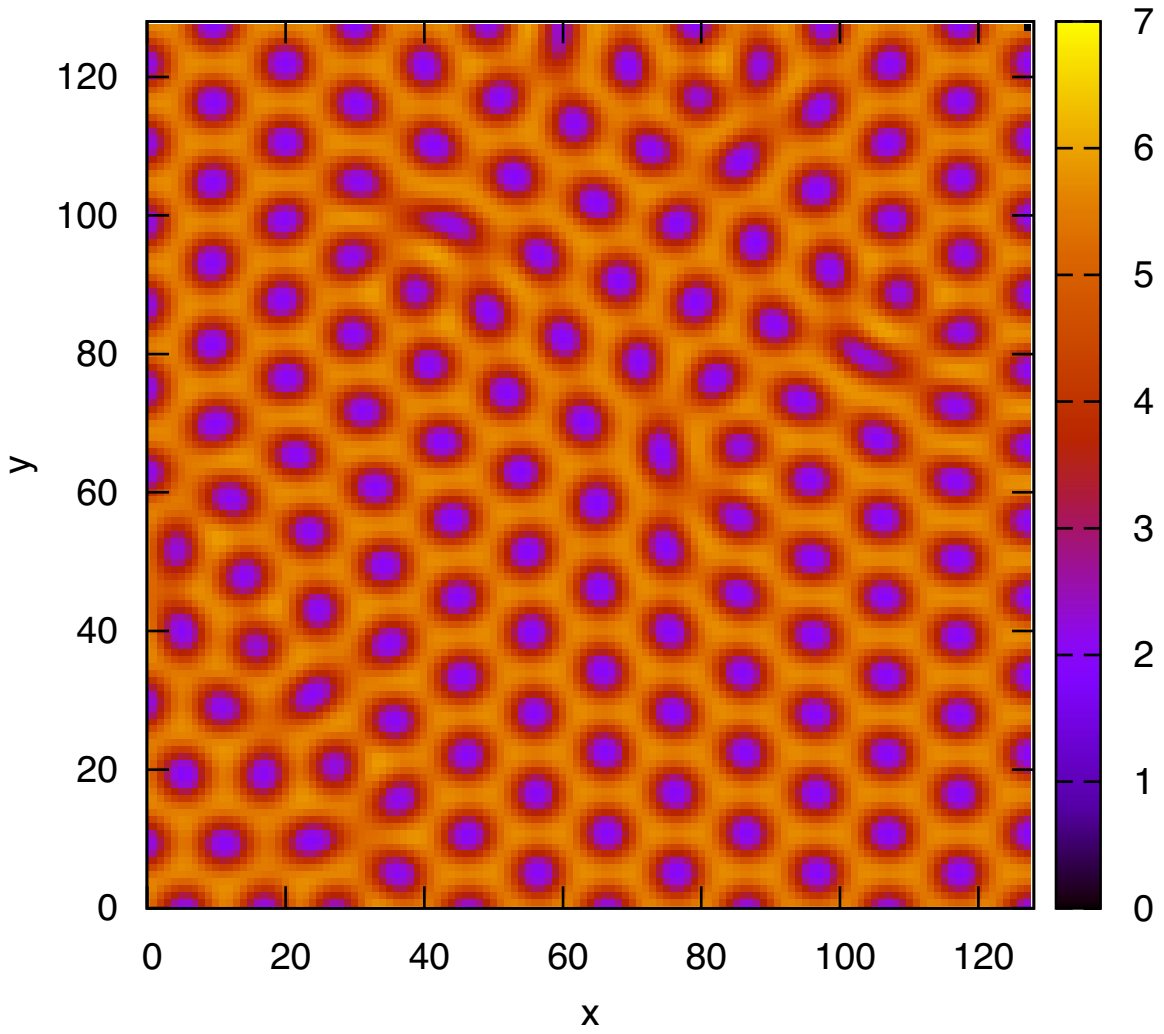
EX2: SOME EXPERIMENTS

- **reactions**
  - -> *version of Brusselator model, http://en.wikipedia.org/wiki/Brusselator*
- **parameters**
  - -> *Pena, Perez-Garcia: Stability of Turing patterns in the Brusselator model, Physical Review 2001*
  - -> *a = 4.5, b = 0.04 ..0.98, D = 128, h = 0.8*
- **unfolding**
  - -> *runtime (constraint solver, 4 threads): 128 sec*
  - -> *places: 32,768, transitions: 324,616*
- **continues simulation**
  - -> *BDF (Backward Differentiation Formulae, higher-order stiffly stable solver)*
  - -> *simulation time: 5,000 -> runtime: about 30h*

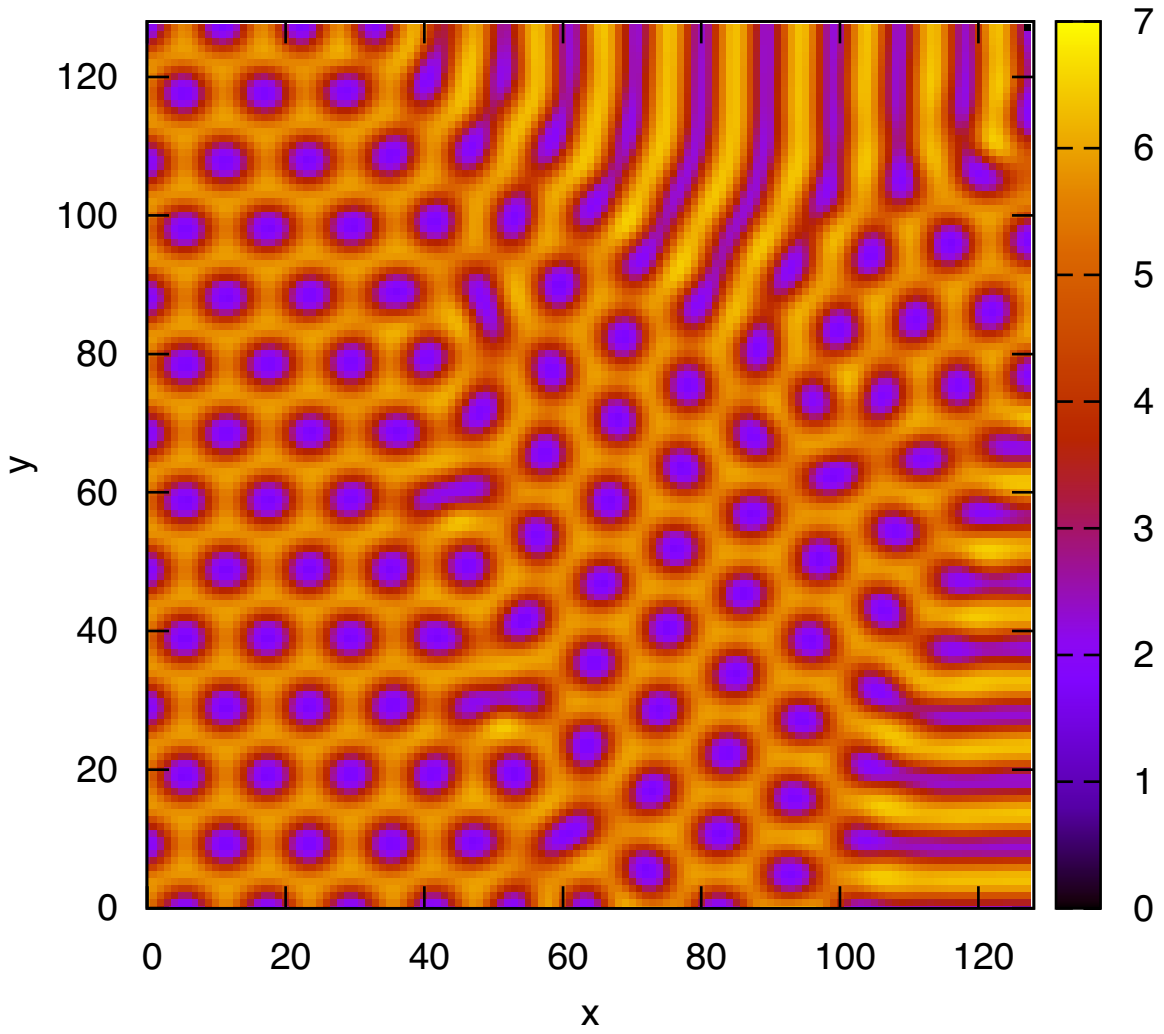
EX2: TRACES


7.80
6.50
5.20
3.90
2.60
1.30
0.00
0.00
10.00
20.00
30.00
40.00
50.00
60.00
70.00
80.00
90.00
Time

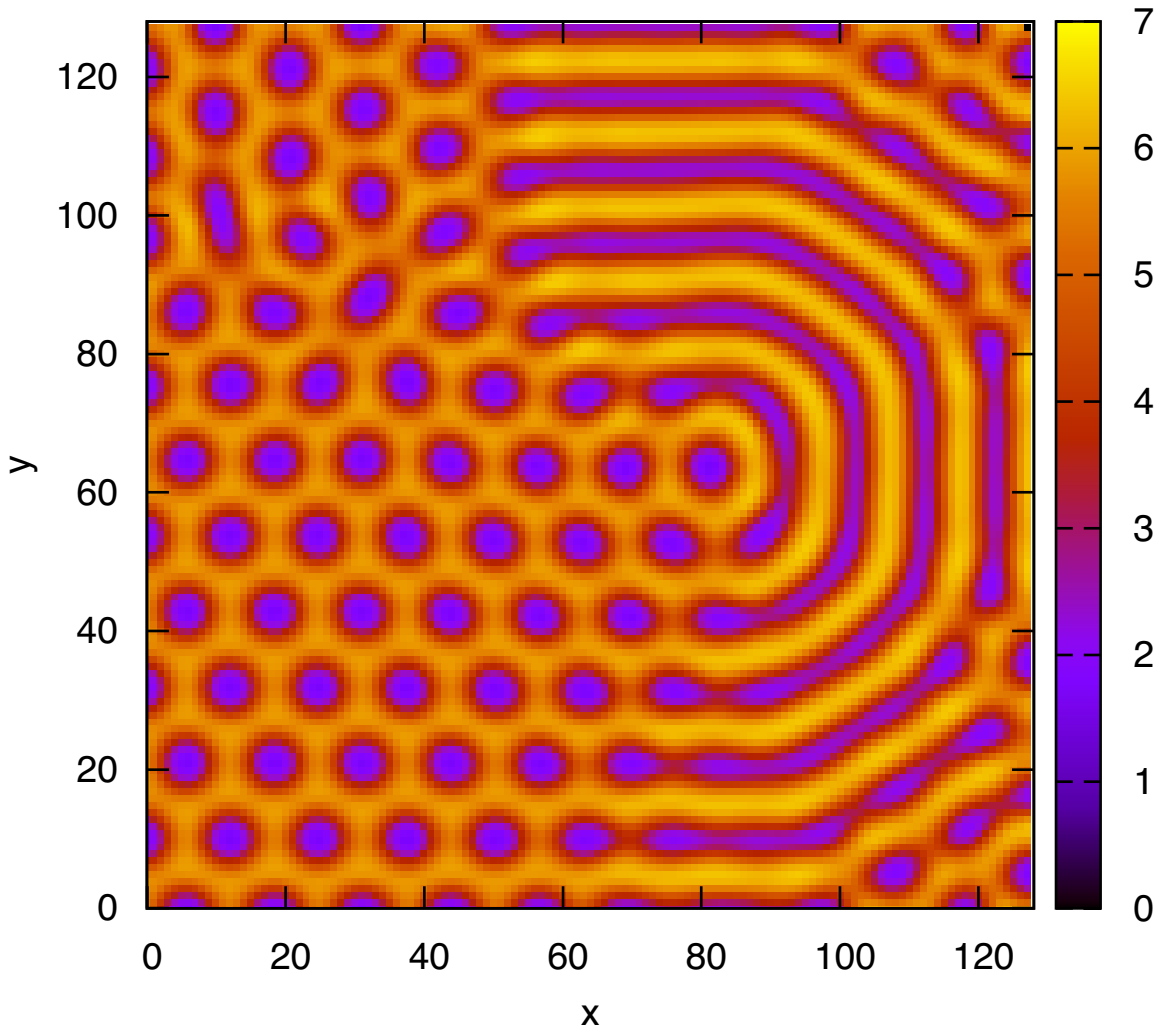
EX2: TURING PATTERNS



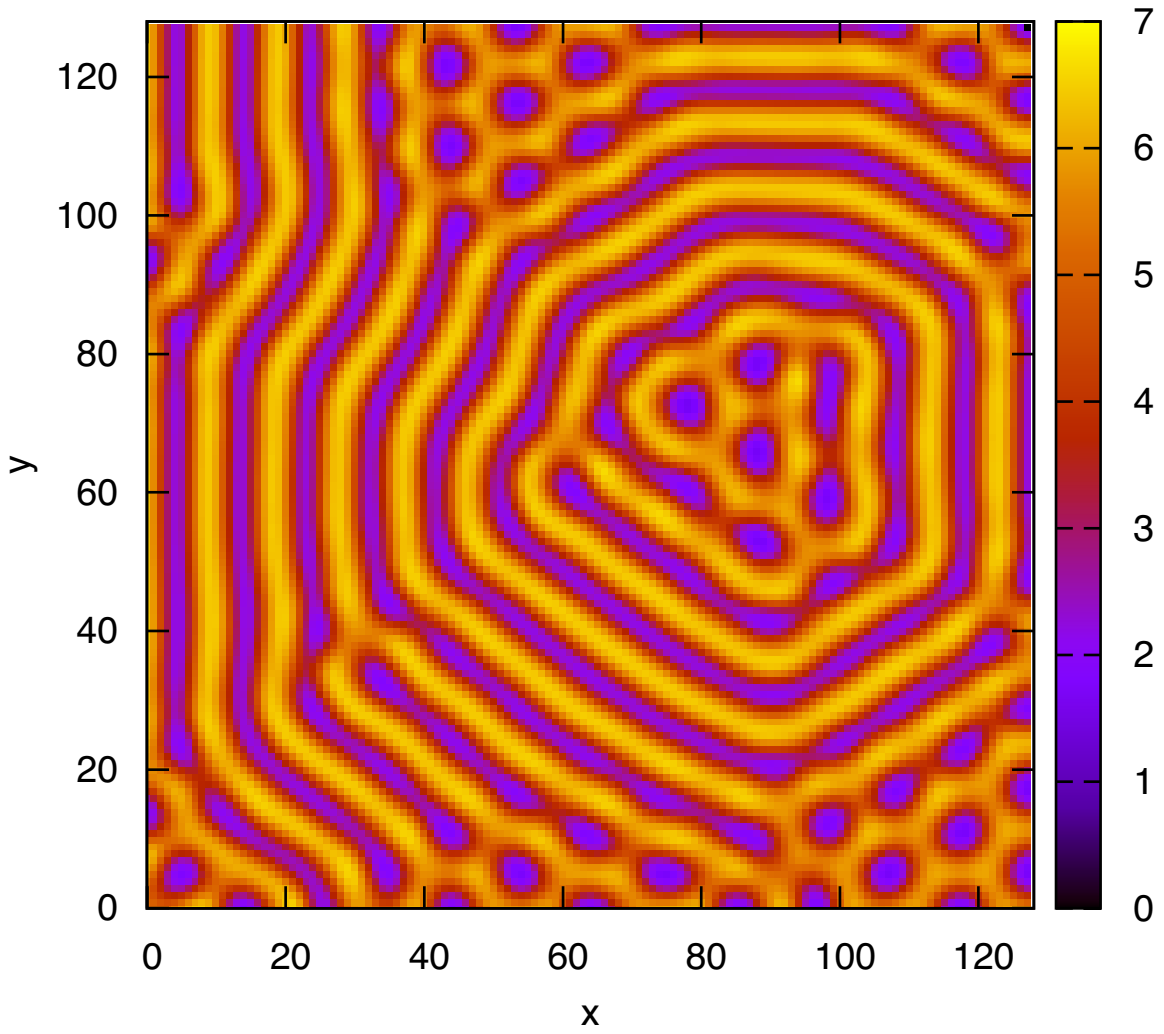
EX2: TURING PATTERNS



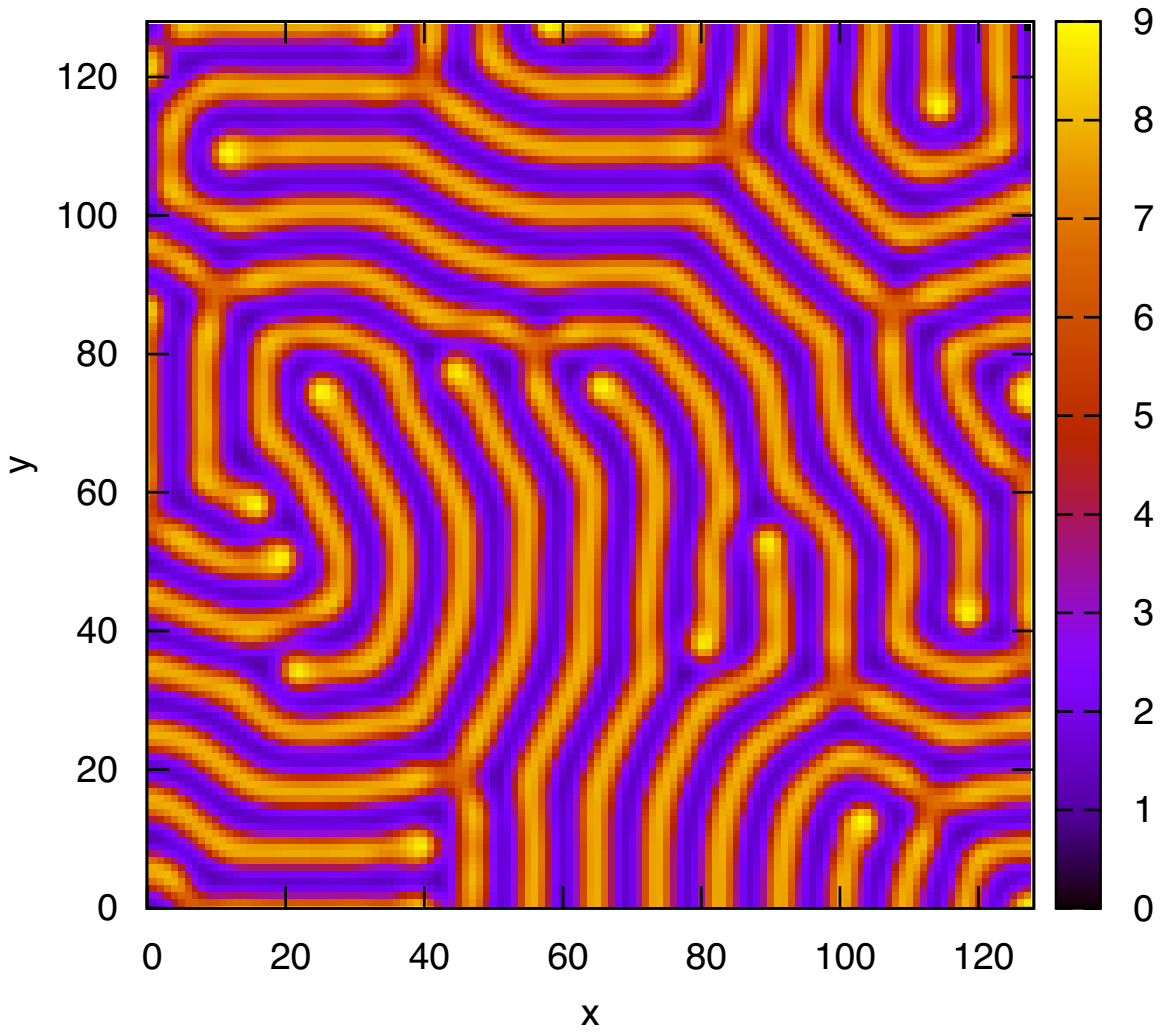
EX2: TURING PATTERNS



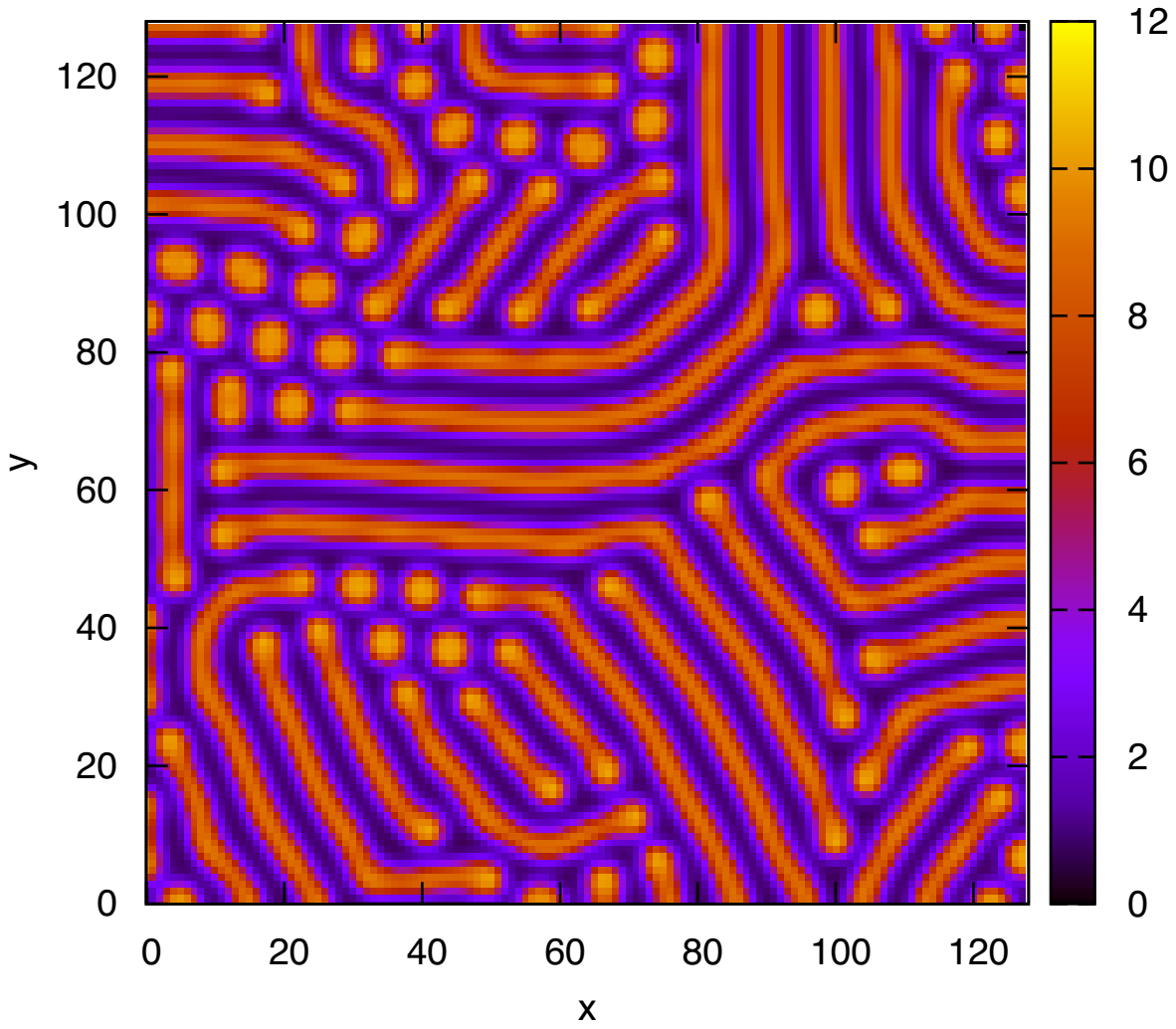
EX2: TURING PATTERNS



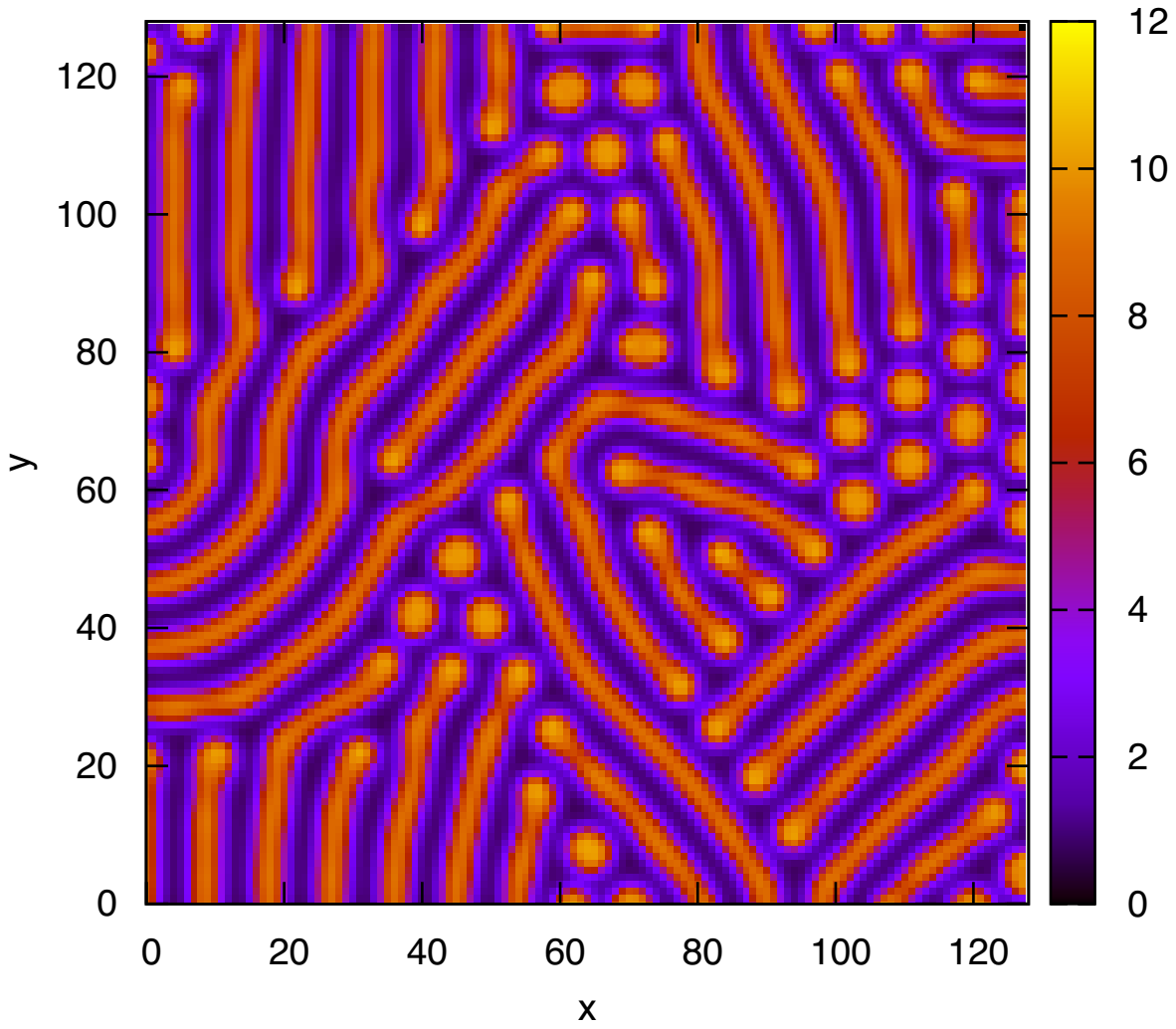
EX2: TURING PATTERNS



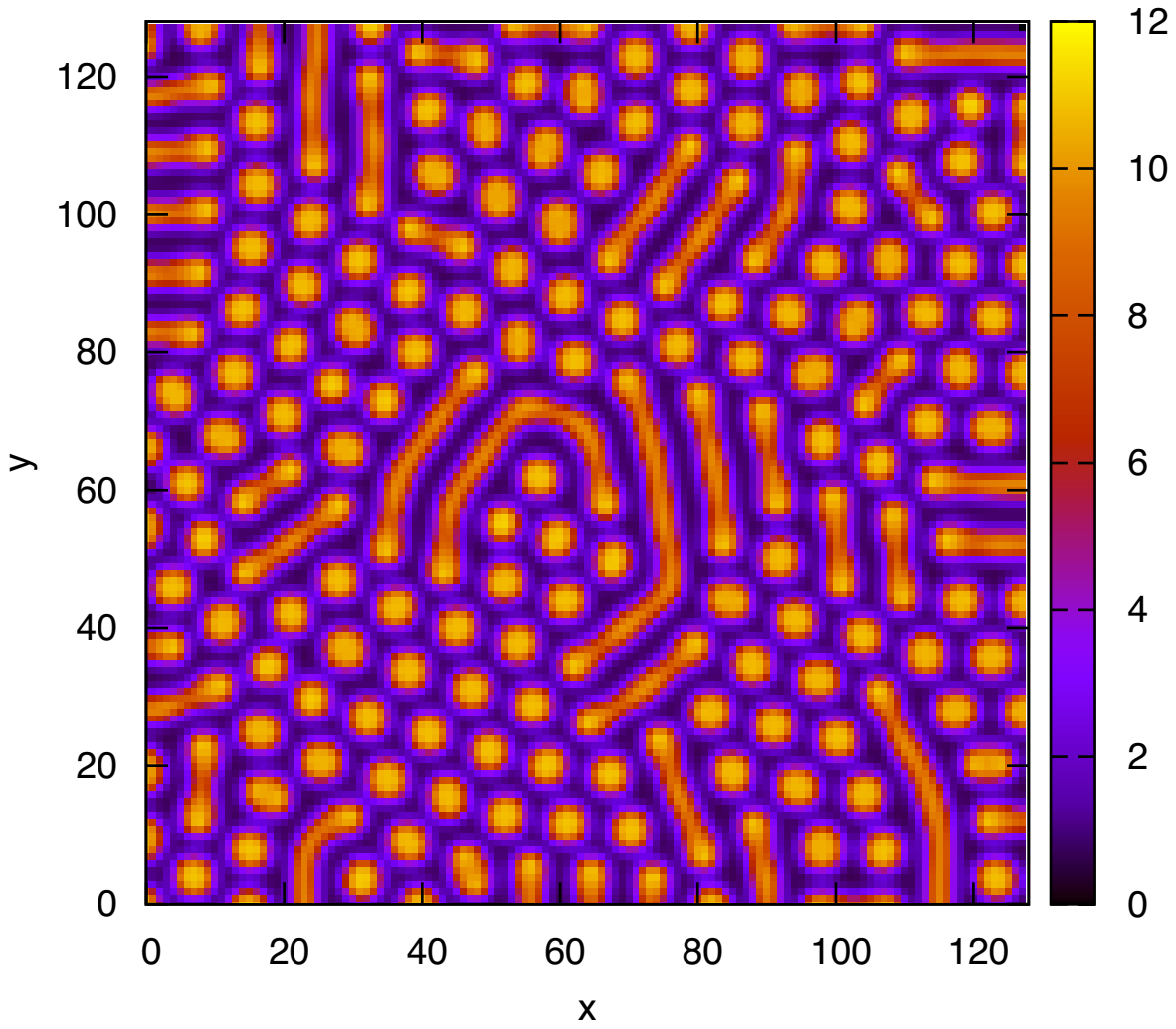
EX2: TURING PATTERNS



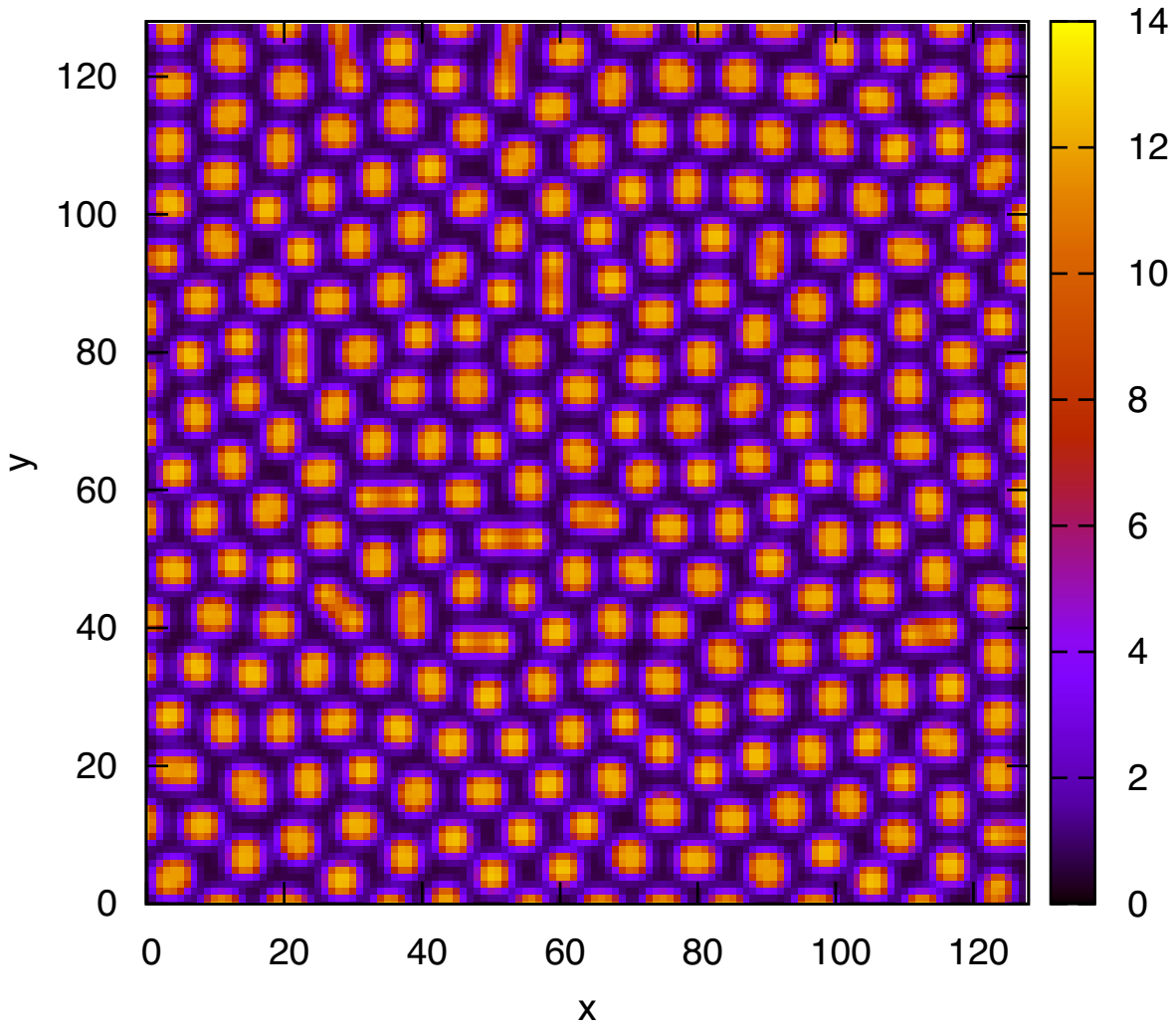
EX2: TURING PATTERNS



EX2: TURING PATTERNS



EX2: TURING PATTERNS



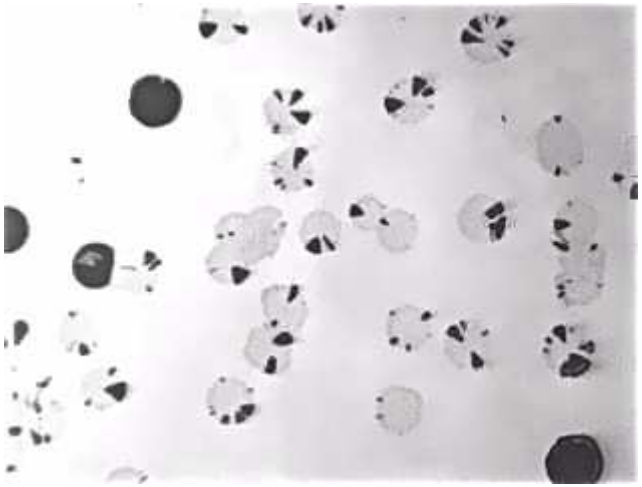
EXAMPLE 3:

PHASE VARIATION IN MULTISTRAIN CELL COLONIES

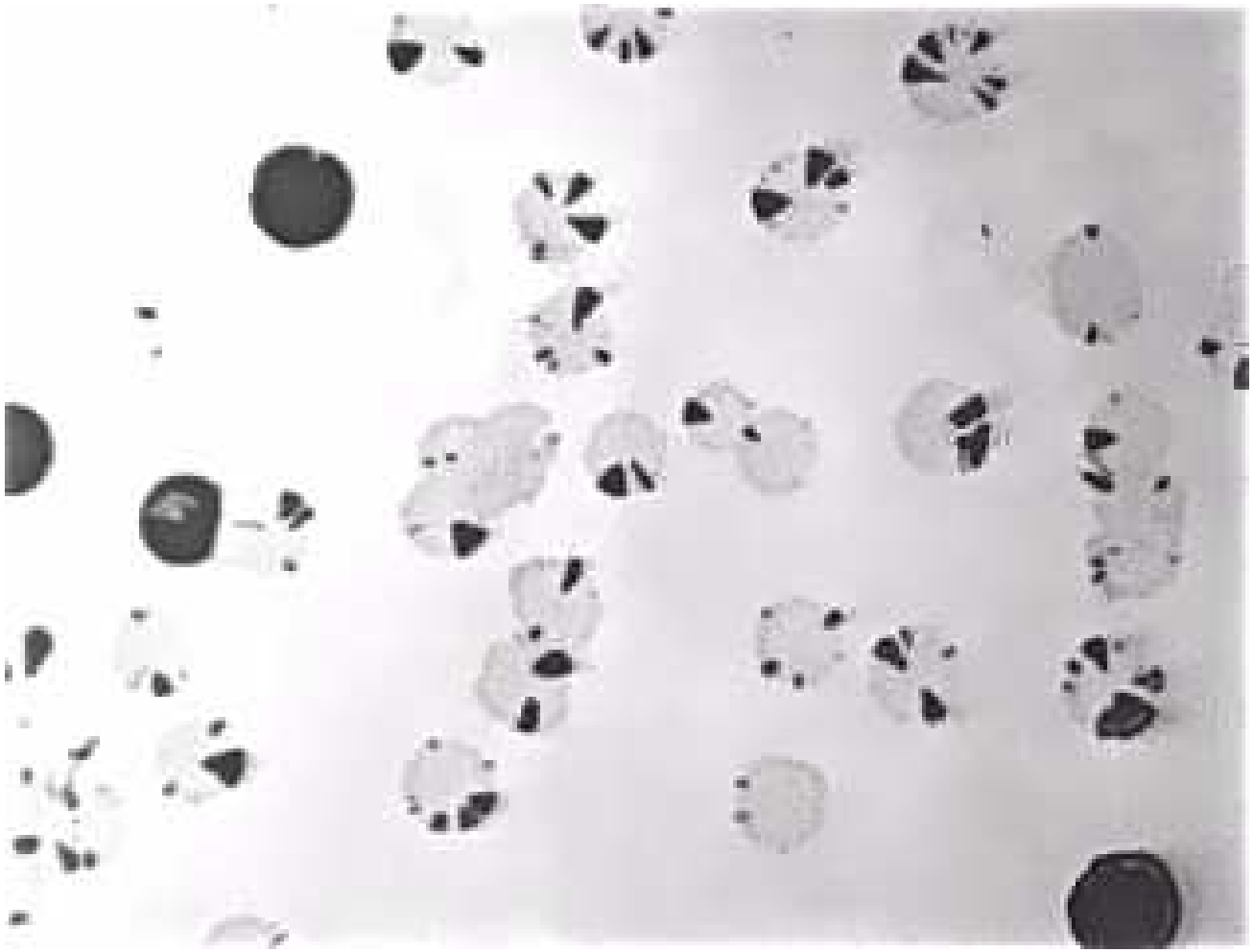
monika.heiner@b-tu.de

EX3: PHASE VARIATION IN CELL COLONIES

- **phase variation**
  - -> *method for dealing with rapidly varying environments without requiring random mutations*
- **contingency genes**
  - -> *populations include variants adapted to "foreseeable" frequently encountered environmental or selective conditions*
- **stochastic gene switching process**
  - -> *controlled by reversible gene mutations, inversions, or epigenetic modification*
  - -> *e.g. switch between two phenotypes A, B*
- **colonial sectoring**
  - -> *observable effect in cultures grown in vitro*



EX3: CELL COLONIES, WETLAB OBSERVATIONS



(courtesy of N Saunders)

Microbiology (2003), **149**, 485–495

DOI 10.1099/mic.0.25807-0

Mutation rates: estimating phase variation rates when fitness differences are present and their impact on population structure

Nigel J. Saunders,[1]† E. Richard Moxon[1] and Mike B. Gravenor[2]

[1]Molecular Infectious Diseases Group, Institute of Molecular Medicine, University of Oxford, Headington, Oxford OX3 9DS, UK

[2]Institute for Animal Health, Compton, Berkshire RG20 7NN, UK

Correspondence
Nigel J. Saunders
saunders@molbiol.ox.ac.uk

Phase variation is a mechanism of ON–OFF switching that is widely utilized by bacterial pathogens. There is currently no standardization to how the rate of phase variation is determined experimentally.

Microbiology (2003), **149**, 485–495

DOI 10.[illegible]/mic.0.25807-0

Mutation rates: estimati[illegible] phas[illegible] variation [illegible]s when fitness diff[illegible]nce[illegible] a[illegible] pre[illegible]nt a[illegible]d their impact on po[illegible]latio[illegible] st[illegible]

Nigel J.[illegible]1† E[illegible]ichard M[illegible]xon[1] and Mike B. Gravenor[2]

[1]Molecular Inf[illegible]ious D[illegible]ases Group, Institute of Molecular Medicine, University of Oxford, [illegible]eadington, O[illegible]X3 9DS, UK

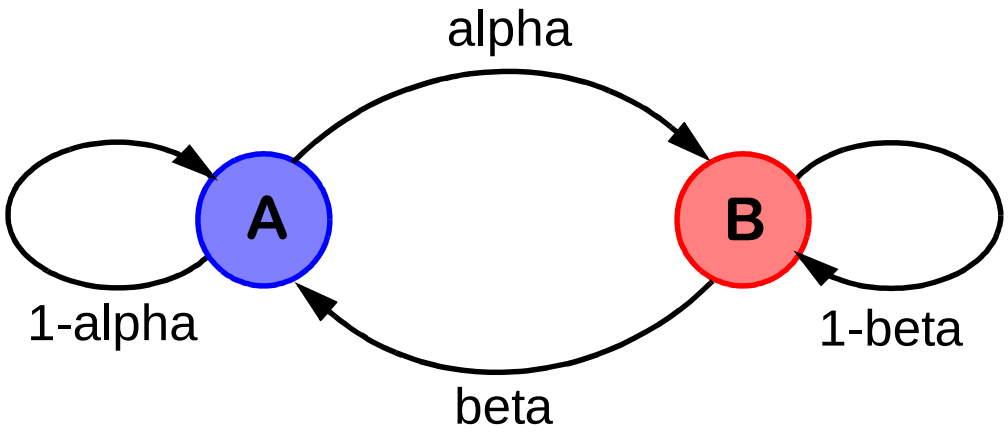
[2]Institute for Animal Health, Compton, Berkshire RG20 7NN, UK

Correspondence
Nigel J. S[illegible]nders
[illegible]nders@[illegible]biol.o[illegible]uk

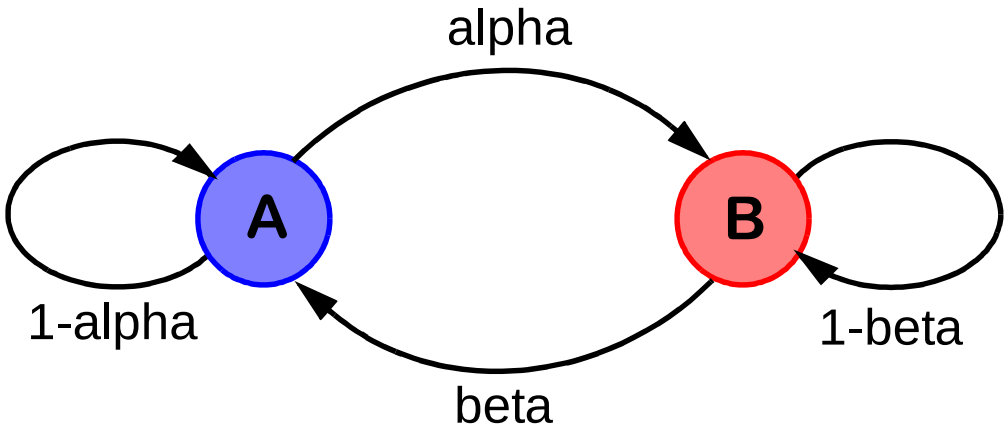
Phase variation is a mechanism of ON–OFF switching that is widely utilized by bacterial pathogens. There is currently no standardization to how the rate of phase variation is determined experimentally.

NO SPACE

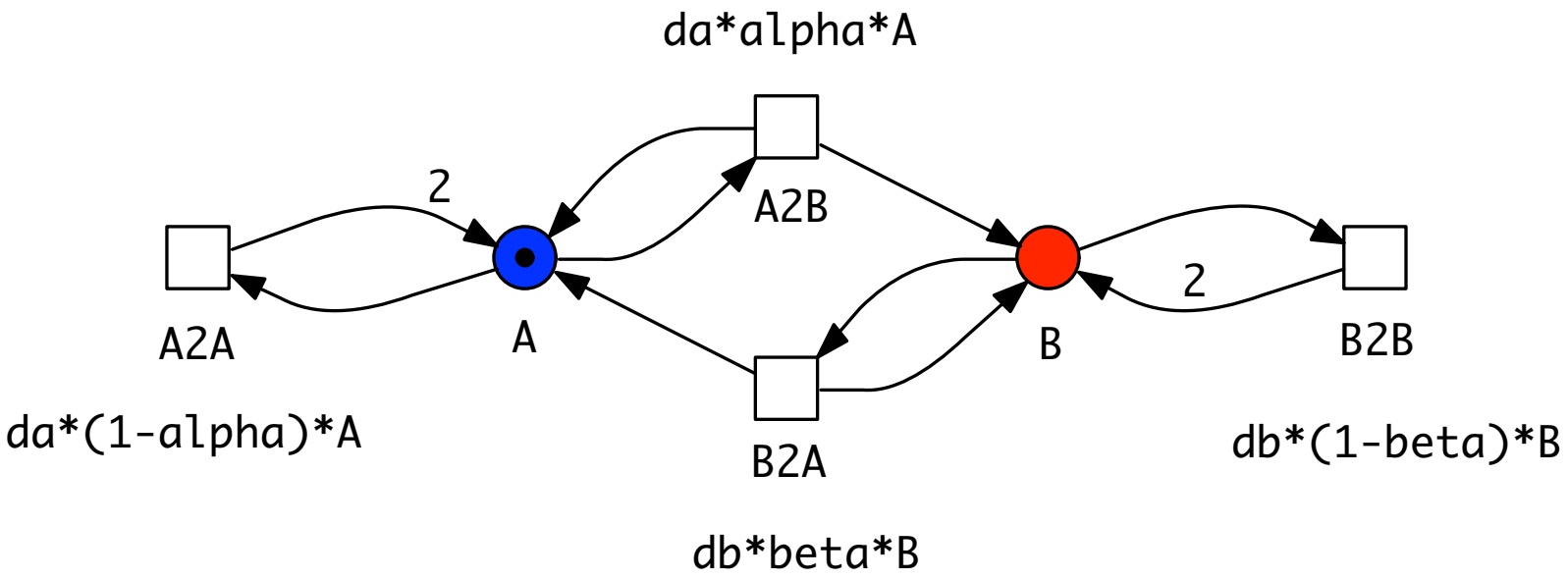
- **two cell types: phenotype A and B**
- **cell divide**
  - -> *cell division may involve mutation of the offspring*
  - -> *parent cell keeps its phenotype*



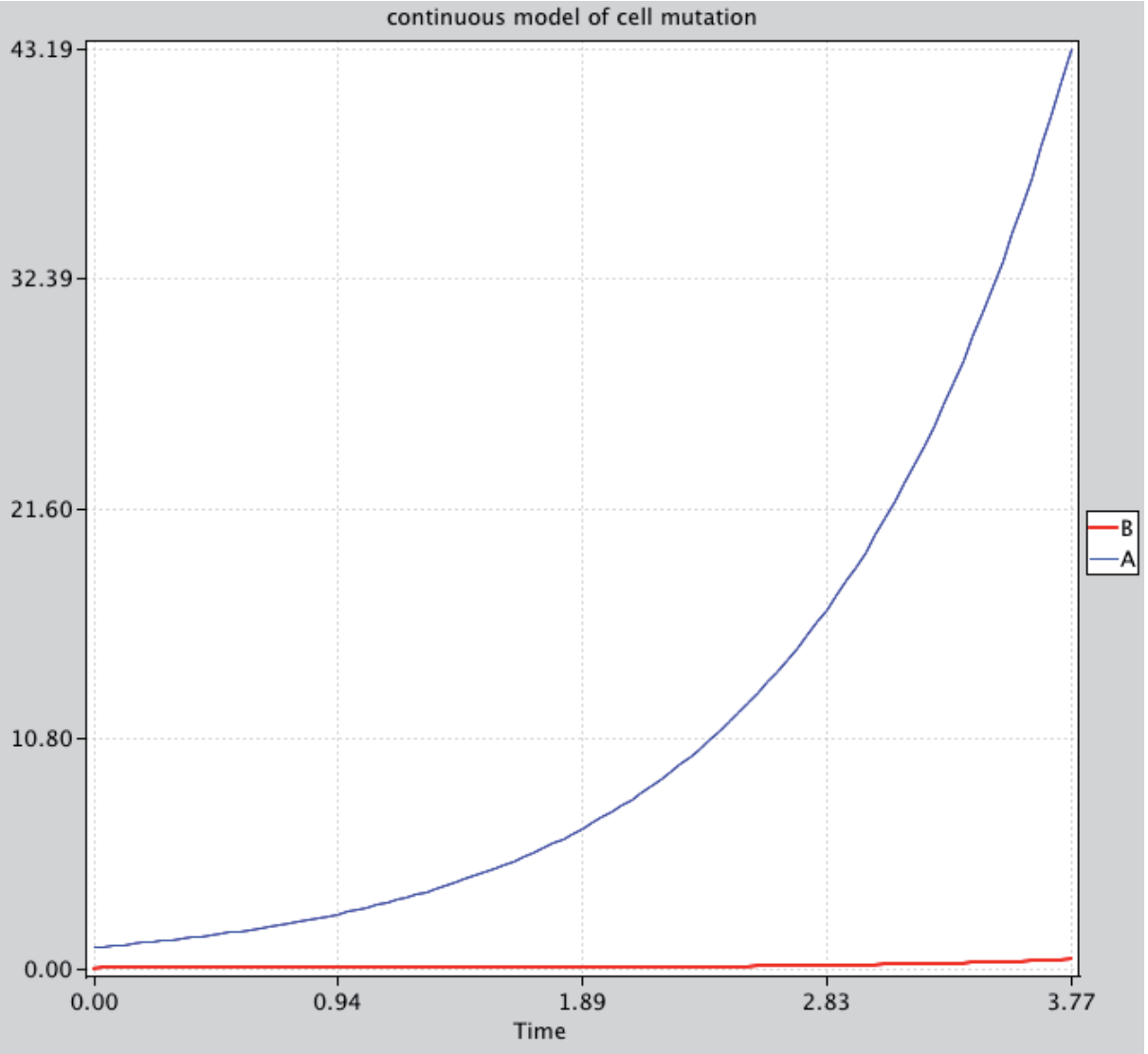
- **two cell types: phenotype A and B**
- **cell divide**
  - -> *cell division may involve mutation of the offspring*
  - -> *parent cell keeps its phenotype*
- **model parameters**
  - -> *alpha = beta - mutation rates*
  - -> *da, db - fitness of A, B*
  - -> *da/db - relative fitness*
- **output**
  - -> *total number of cells*
  - -> *proportion of A = A / (A + B)*
  - -> *proportion of B = B / (A + B)*



EX3: CELL COLONIES, PETRI NET

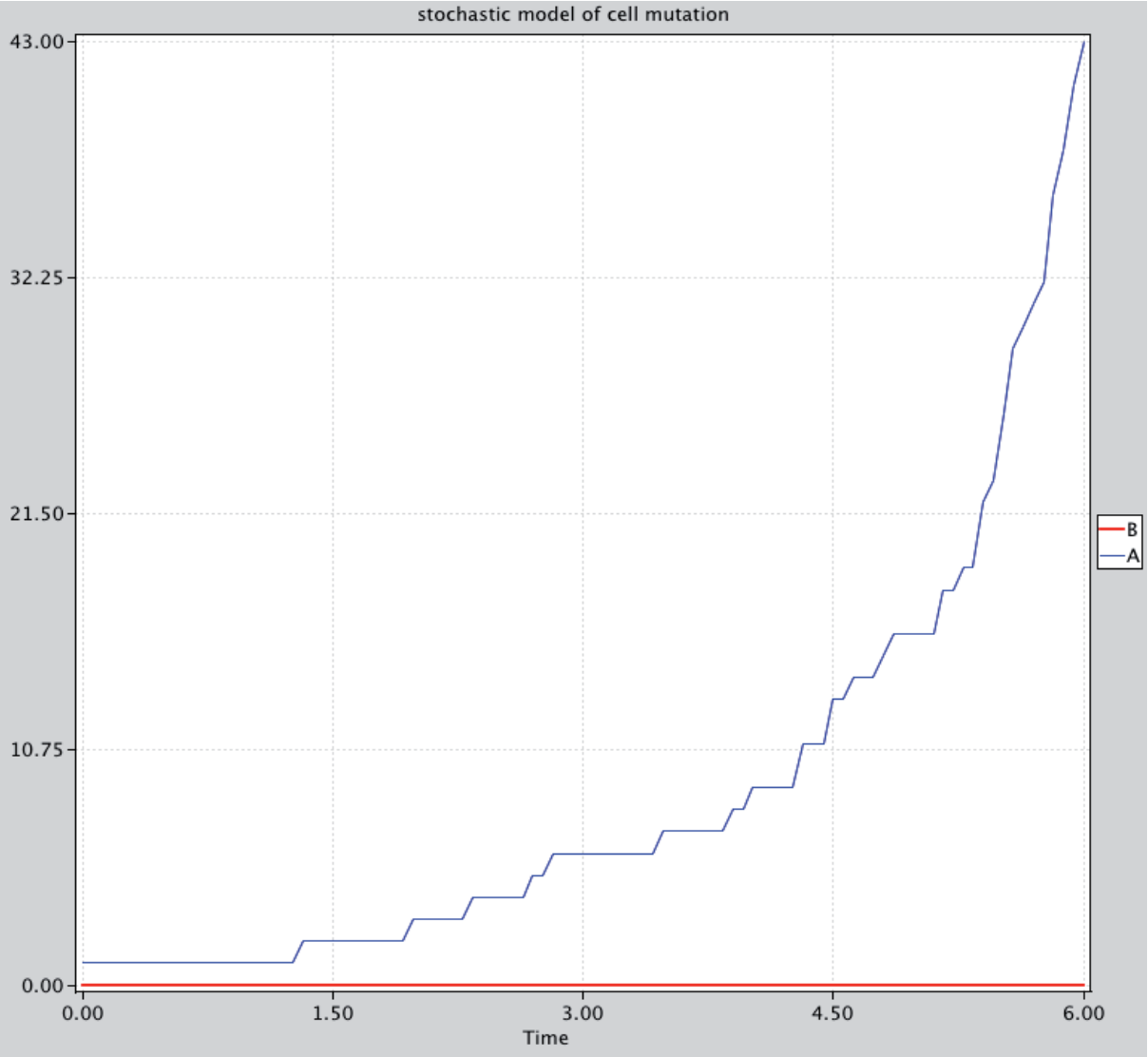


EX3: CELL COLONIES, CONTINUOUS PLOT



monika.heiner@b-tu.de

EX3: CELL COLONIES, STOCHASTIC PLOT

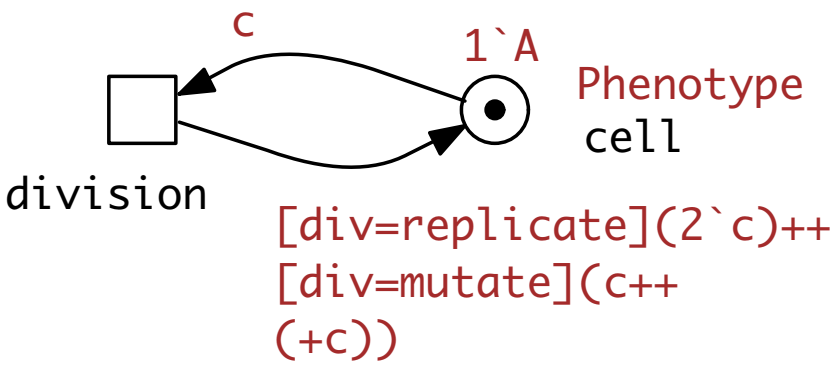


monika.heiner@b-tu.de

colorset Phenotype = **enum with** A, B;
colorset DivisionType = **enum with** replicate , mutate ;

EX3: CELL COLONIES, BASIC MODEL

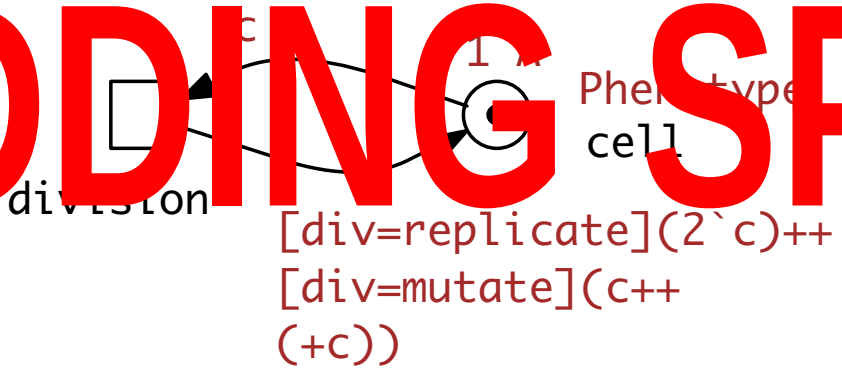
colorset Phenotype = **enum with** A, B;
colorset DivisionType = **enum with** replicate , mutate ;




```
(c=A) & (div=replicate) : cell*da*(1-alpha)
(c=A) & (div=mutate) : cell*(da*alpha)
(c=B) & (div=replicate) : cell*(db*(1-beta))
(c=B) & (div=mutate) : cell*(db*beta)
```

colorset Phenotype = **enum with** A, B;
colorset DivisionType = **enum with** replicate , mutate ;

ADDING SPACE




```
(c=A) & (div=replicate) : cell*da*(1-alpha)
(c=A) & (div=mutate) : cell*(da*alpha)
(c=B) & (div=replicate) : cell*(db*(1-beta))
(c=B) & (div=mutate) : cell*(db*beta)
```

colorset Phenotype = **enum with** A, B;
colorset DivisionType = **enum with** replicate , mutate ;

ADDING SPACE

CONTROLLING COLONY SPREADING

```
(c=A) & (div=replicate) : cell*da*(1-alpha)
(c=A) & (div=mutate) : cell*(da*alpha)
(c=B) & (div=replicate) : cell*(db*(1-beta))
(c=B) & (div=mutate) : cell*(db*beta)
```

colorset Phenotype = **enum with** A, B;
colorset DivisionType = **enum with** replicate , mutate ;

ADDING SPACE

CONTROLLING COLONY SPREADING

CONTROLLING THICKNESS

colorset Phenotype = **enum with** A, B;
colorset DivisionType = **enum with** replicate , mutate ;

ADDING SPACE

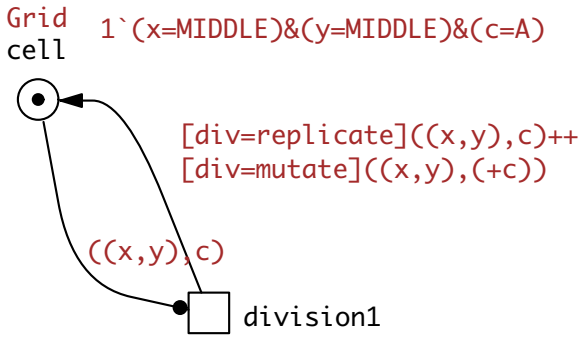
CONTROLLING COLONY SPREADING

CONTROLLING THICKNESS

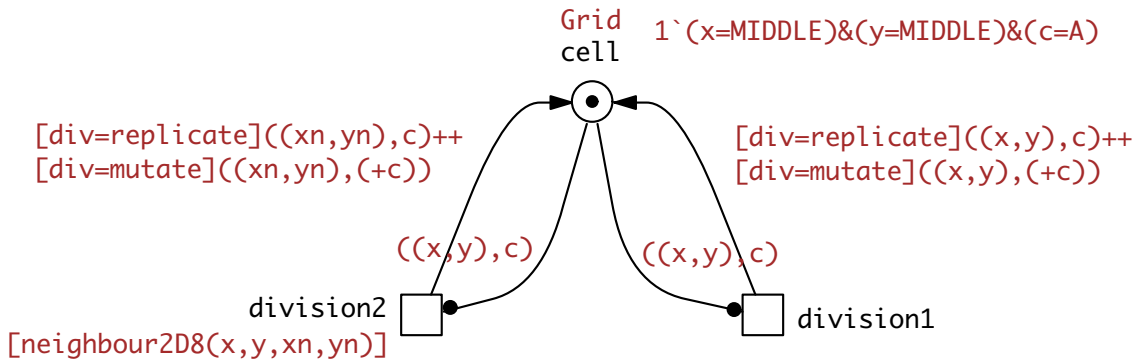
```
(c=A) & (div=replicate) : cell*da*(1-alpha)
(c=A) & (div=mutate)    : cell*(da*alpha)
(c=B) & (div=replicate) : cell*(db*(1-beta))
(c=B) & (div=mutate)    : cell*(db*beta)
```

CONTROLLING COLONY SIZE

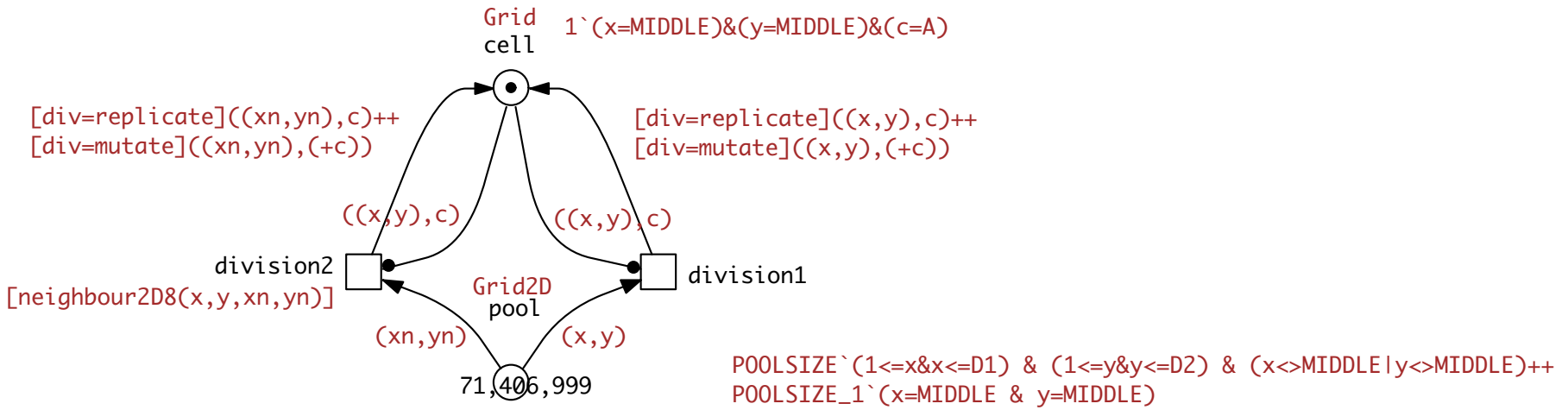
colorset Grid = **product with** Grid2D x Phenotype;



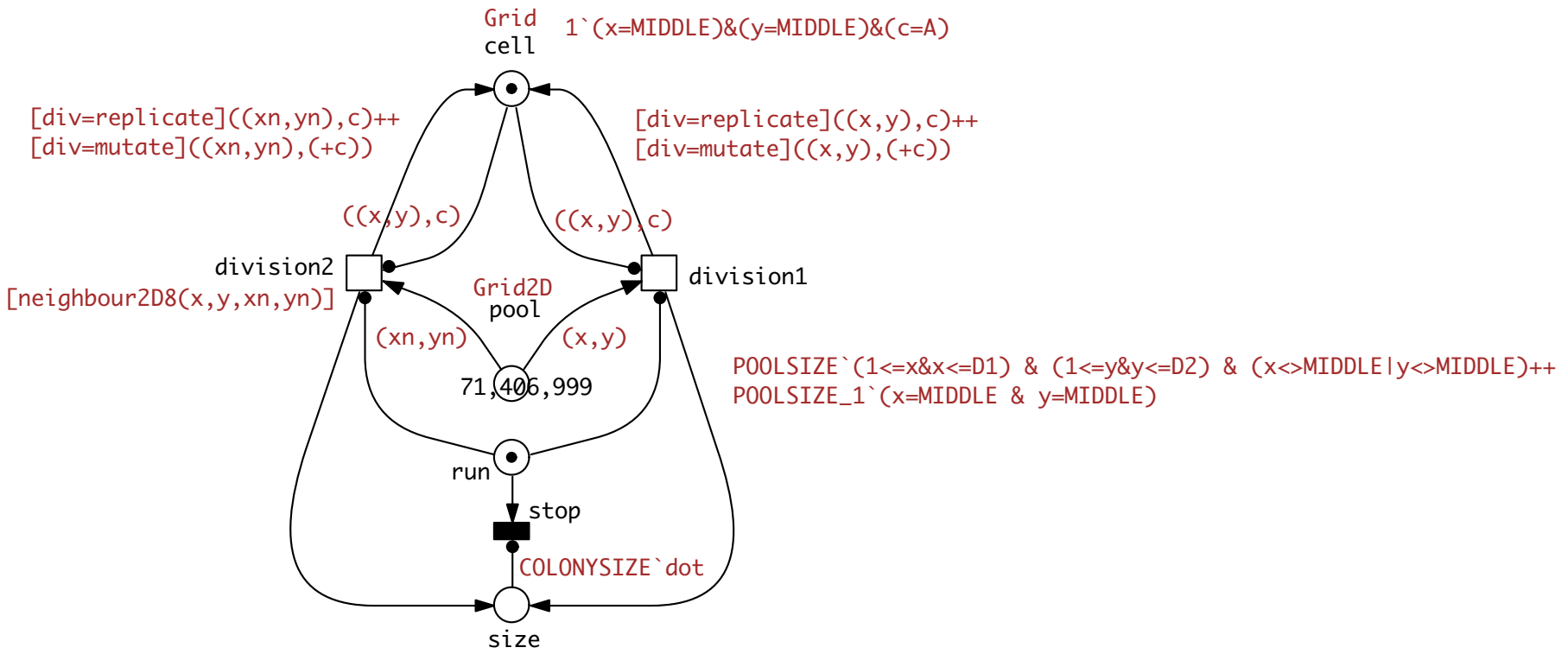
colorset Grid = **product with** Grid2D x Phenotype;



colorset Grid = **product with** Grid2D x Phenotype;



colorset Grid = **product with** Grid2D x Phenotype;

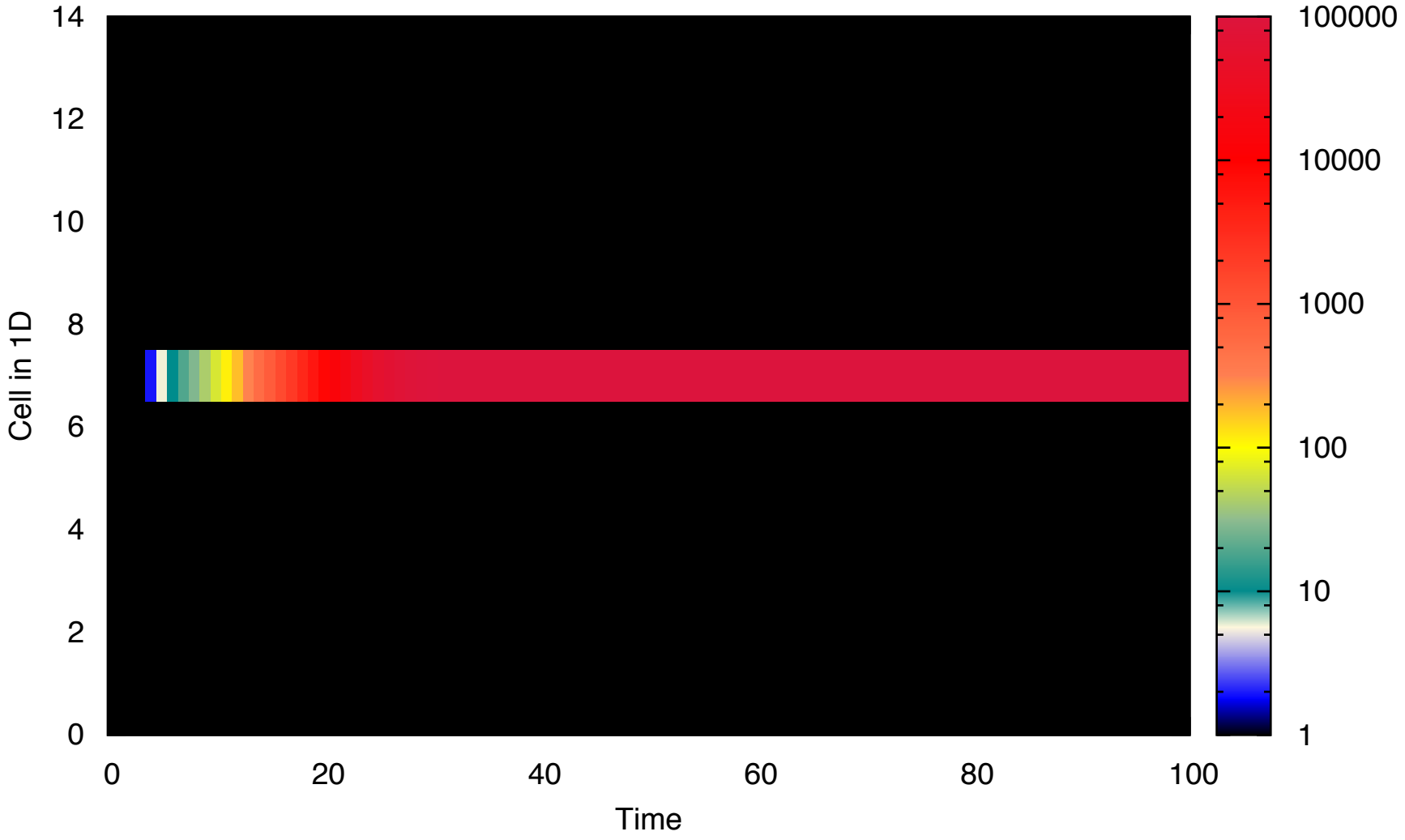


EX3: CELL COLONIES, SOME DETAILS

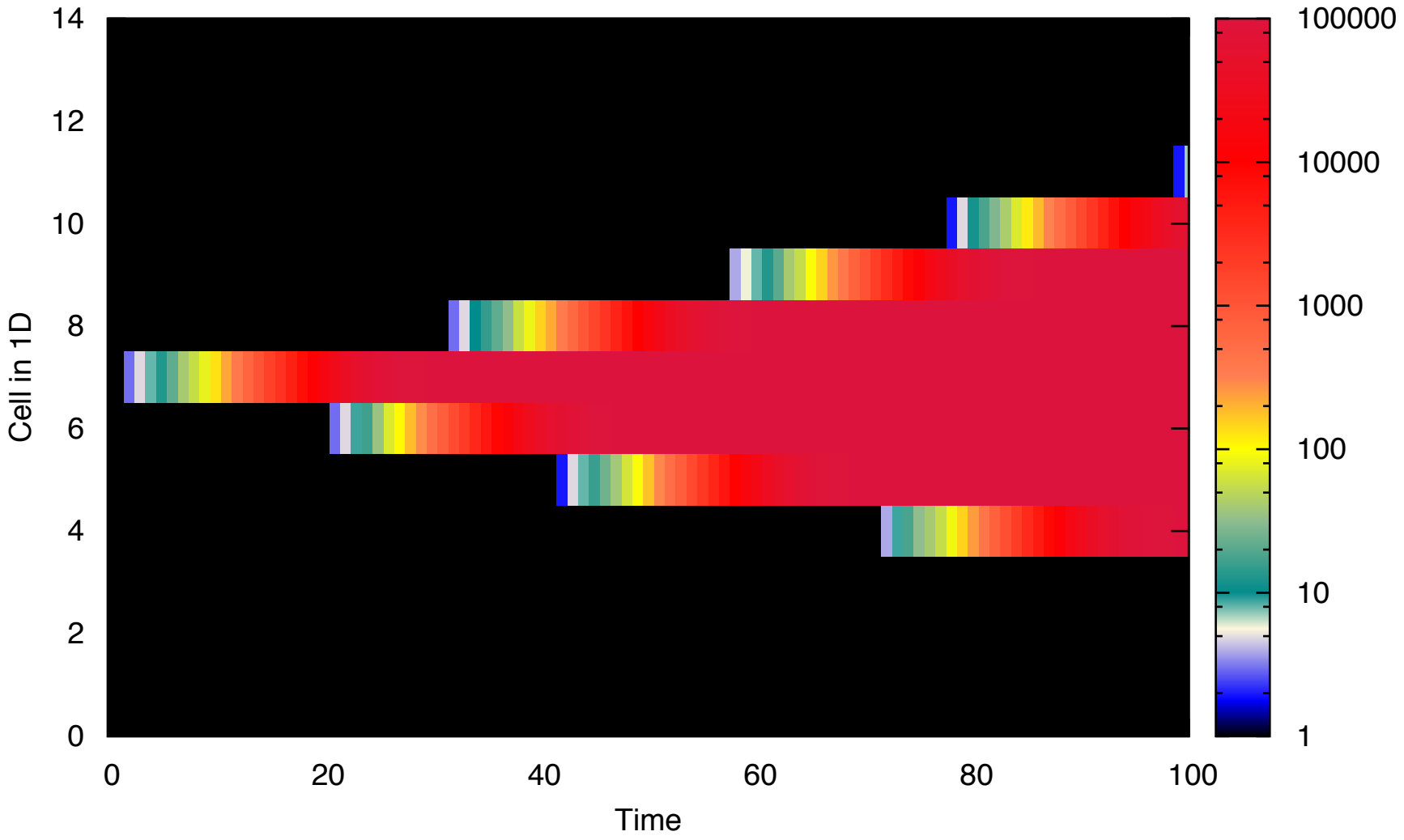
- **model assumptions**
  - -> *"If phase variation occurs, the progeny consists of one A and one B" (Saunders 2003)*
  - -> *It is always the mutant who goes to a neighbouring position, if any.*
  - -> *constant biofilm thickness (so far)*
- **colony size - 24 h**
  - -> *25 generations: 33.5 E+06*
  - -> *26 generations: 67 E+06*
  - -> *COLONYSIZE = 70,000,000*
- **grid size**
  - -> *61 x 61 grid: 11,163 P / 131,044 T; unfolding: 152 sec;*
  - -> *101 x 101 grid: 30,603 P / 362,404 T; unfolding: 9 min;*
  - -> *runtime 1 stoch. simulation: 35-40 minutes*

. . . SOME EXPERIMENTS

EX3: 1D15 - VARYING MOBILITY, GAMMA = 100

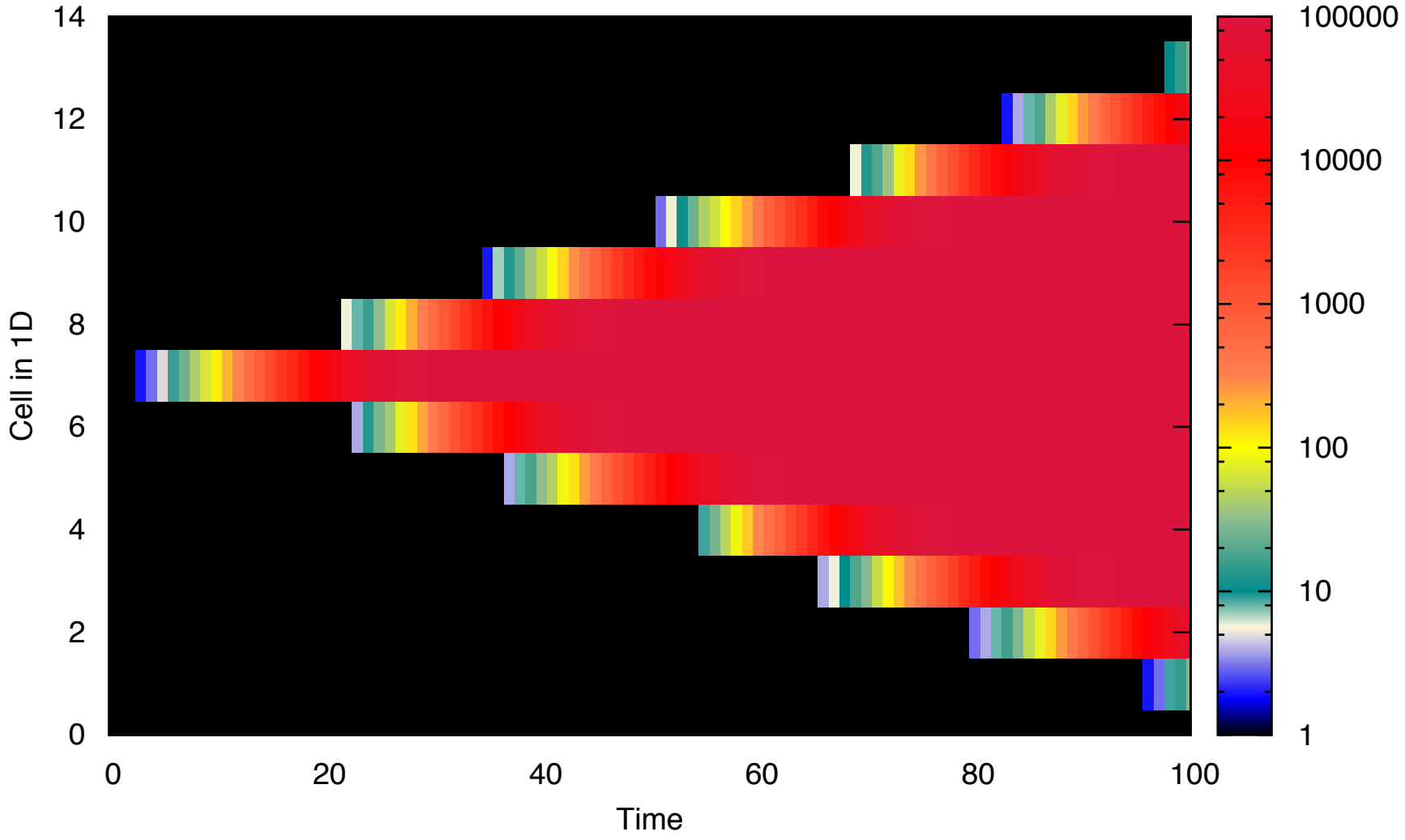


EX3: 1D15 - VARYING MOBILITY, GAMMA = 99.999



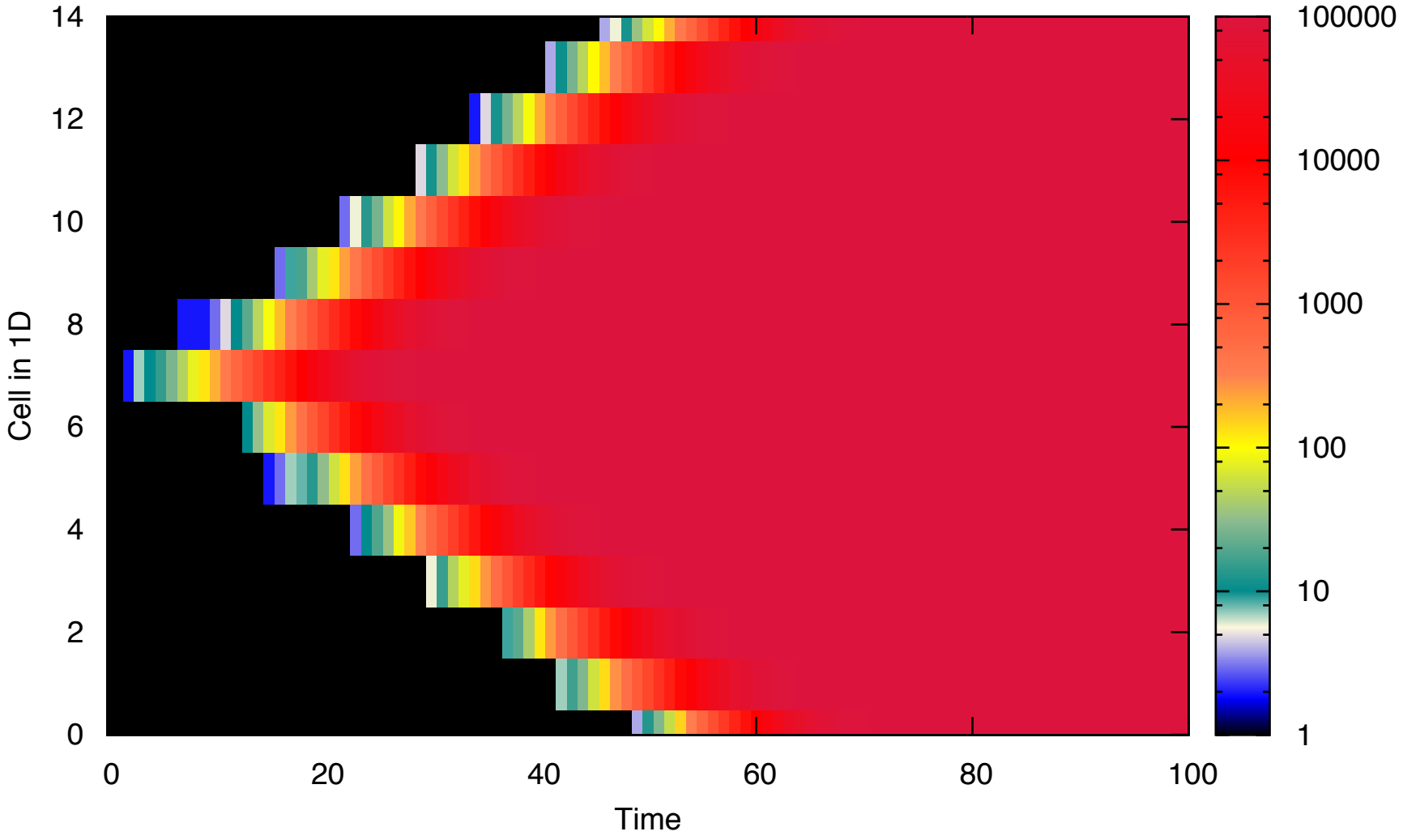
monika.heiner@b-tu.de

Ex3: 1D15 - Varying Mobility, Gamma = 99.99

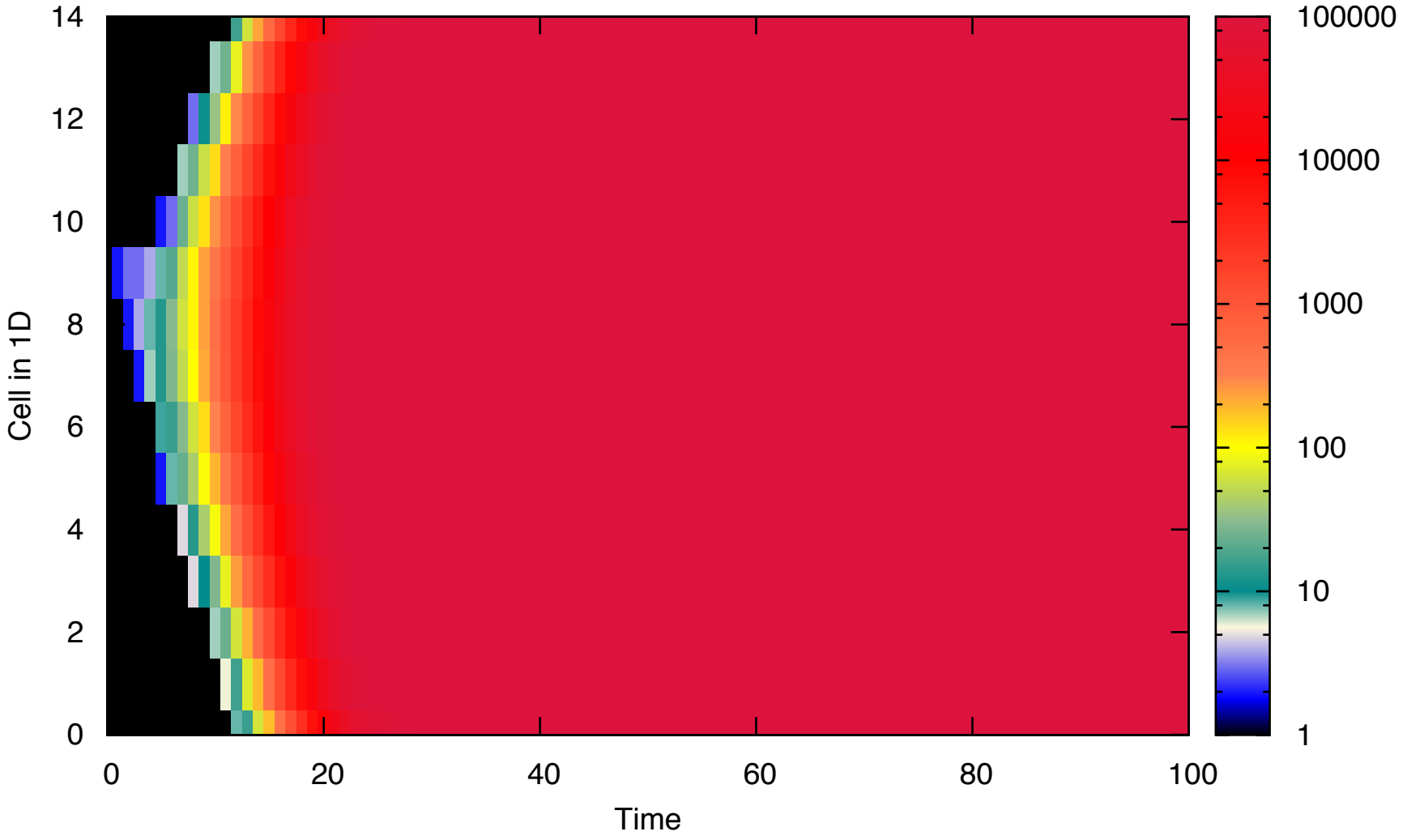


monika.heiner@b-tu.de

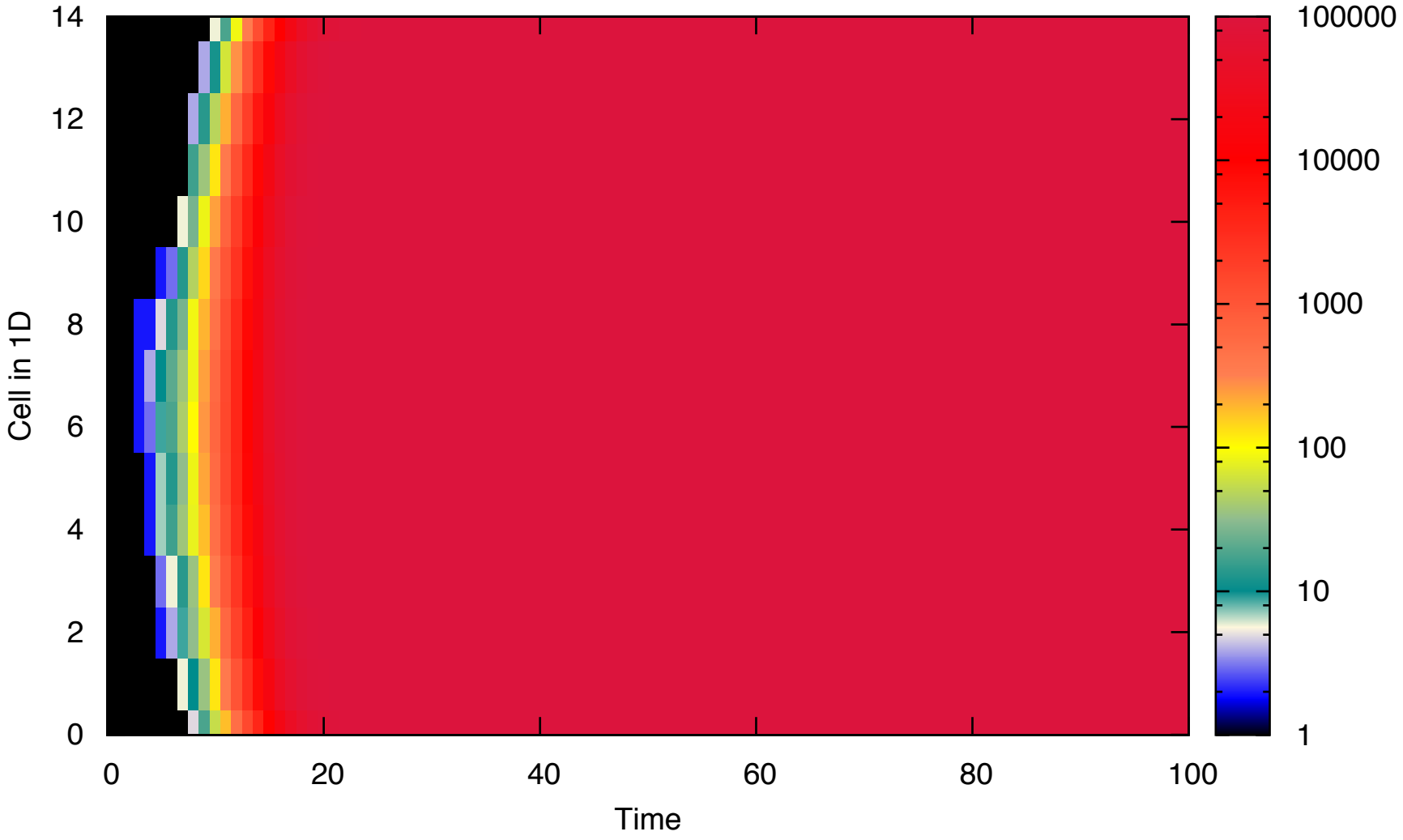
EX3: 1D15 - VARYING MOBILITY, GAMMA = 90



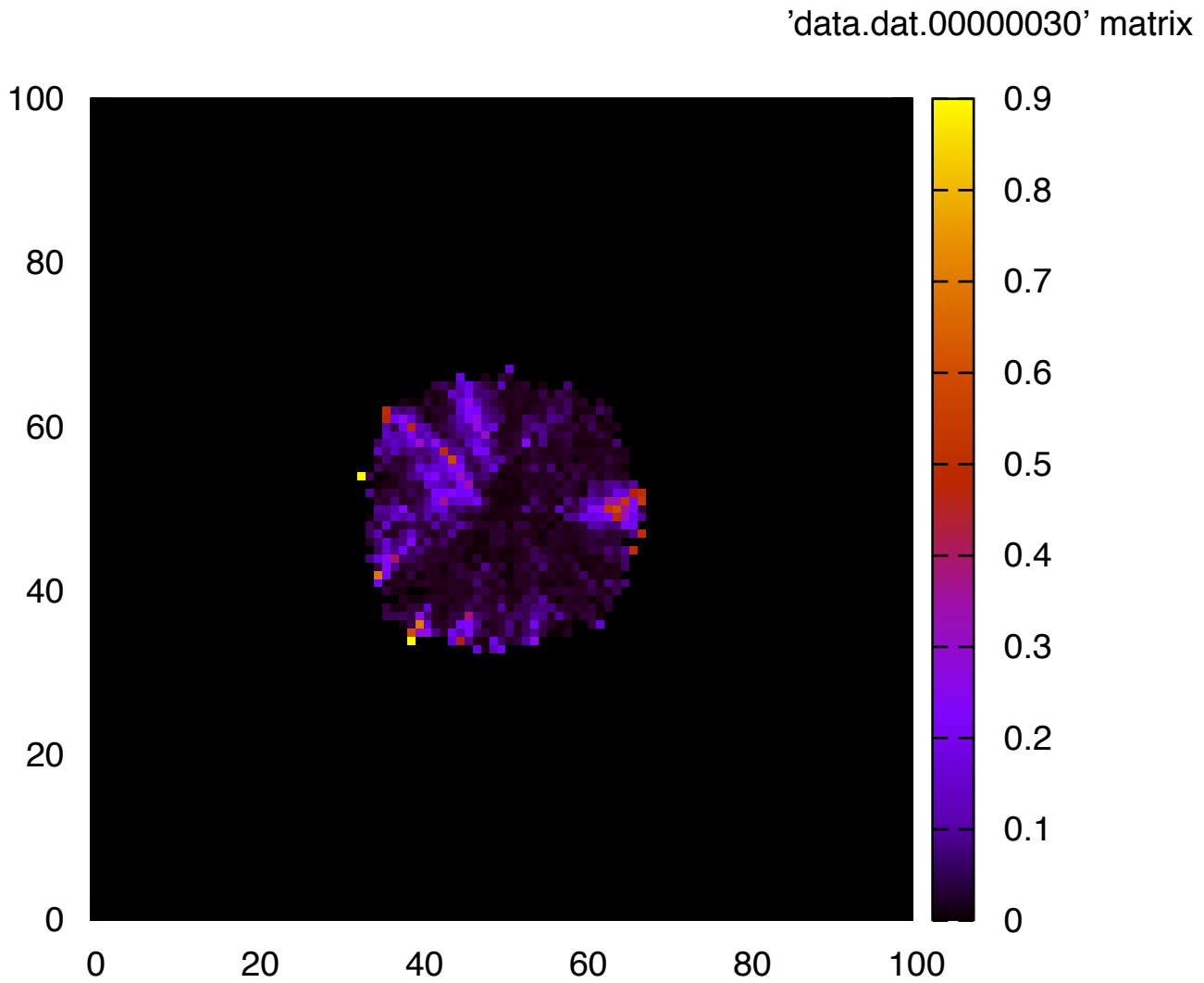
EX3: 1D15 - VARYING MOBILITY, GAMMA = 50



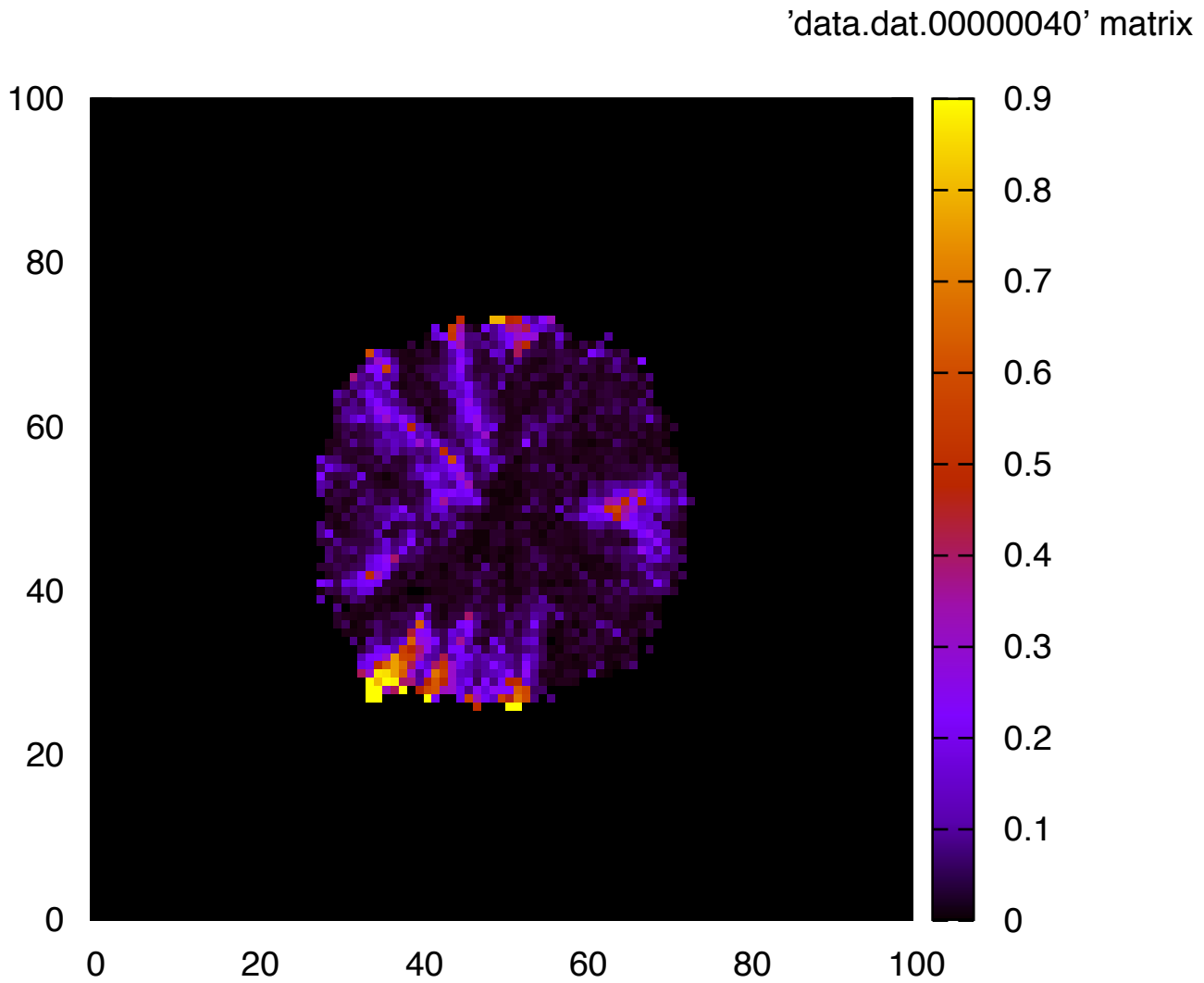
EX3: 1D15 - VARYING MOBILITY, GAMMA = 1



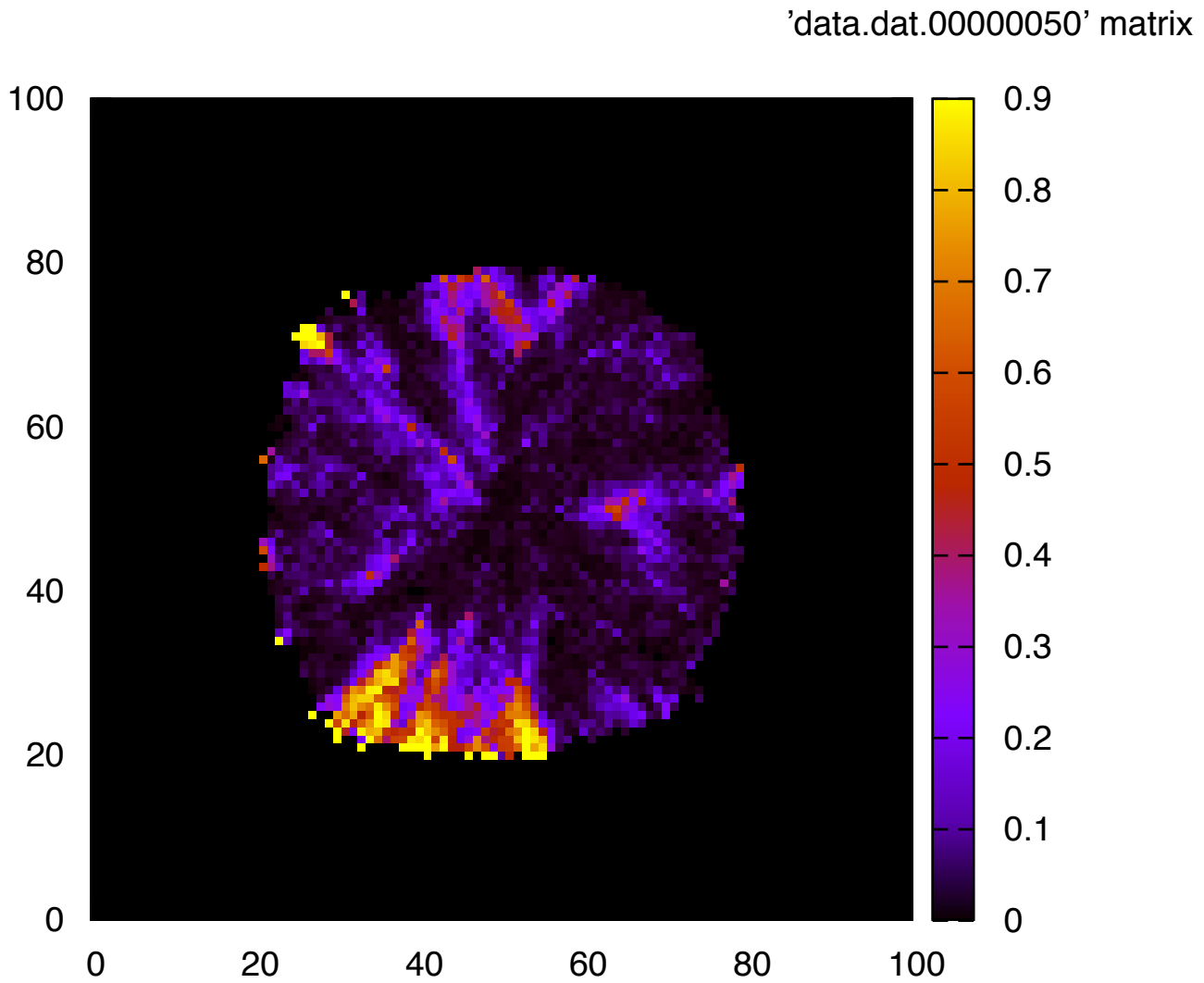
EX3: 2D - TRACE 1 (HIGH, F=1)



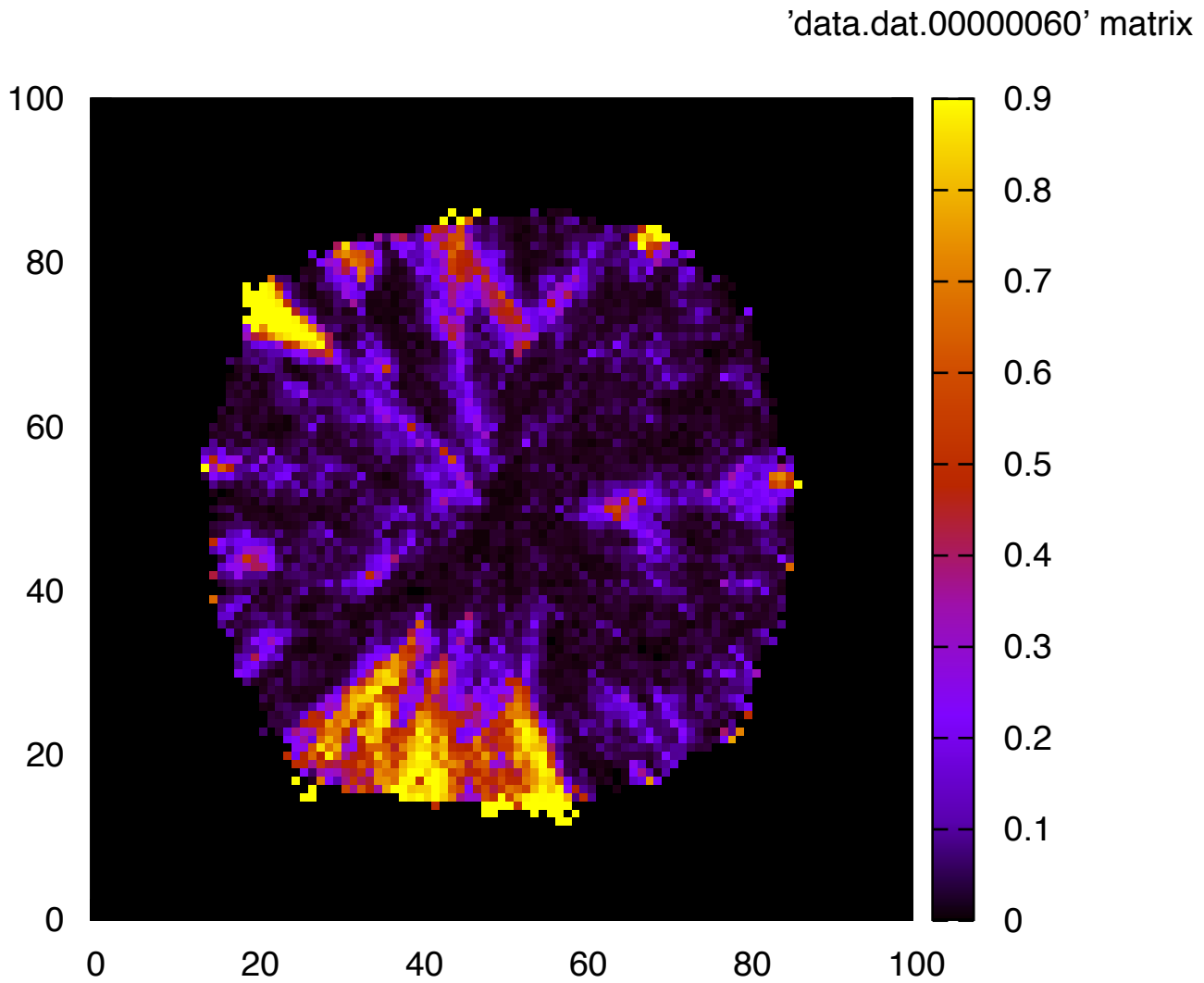
EX3: 2D - TRACE 1 (HIGH, F=1)



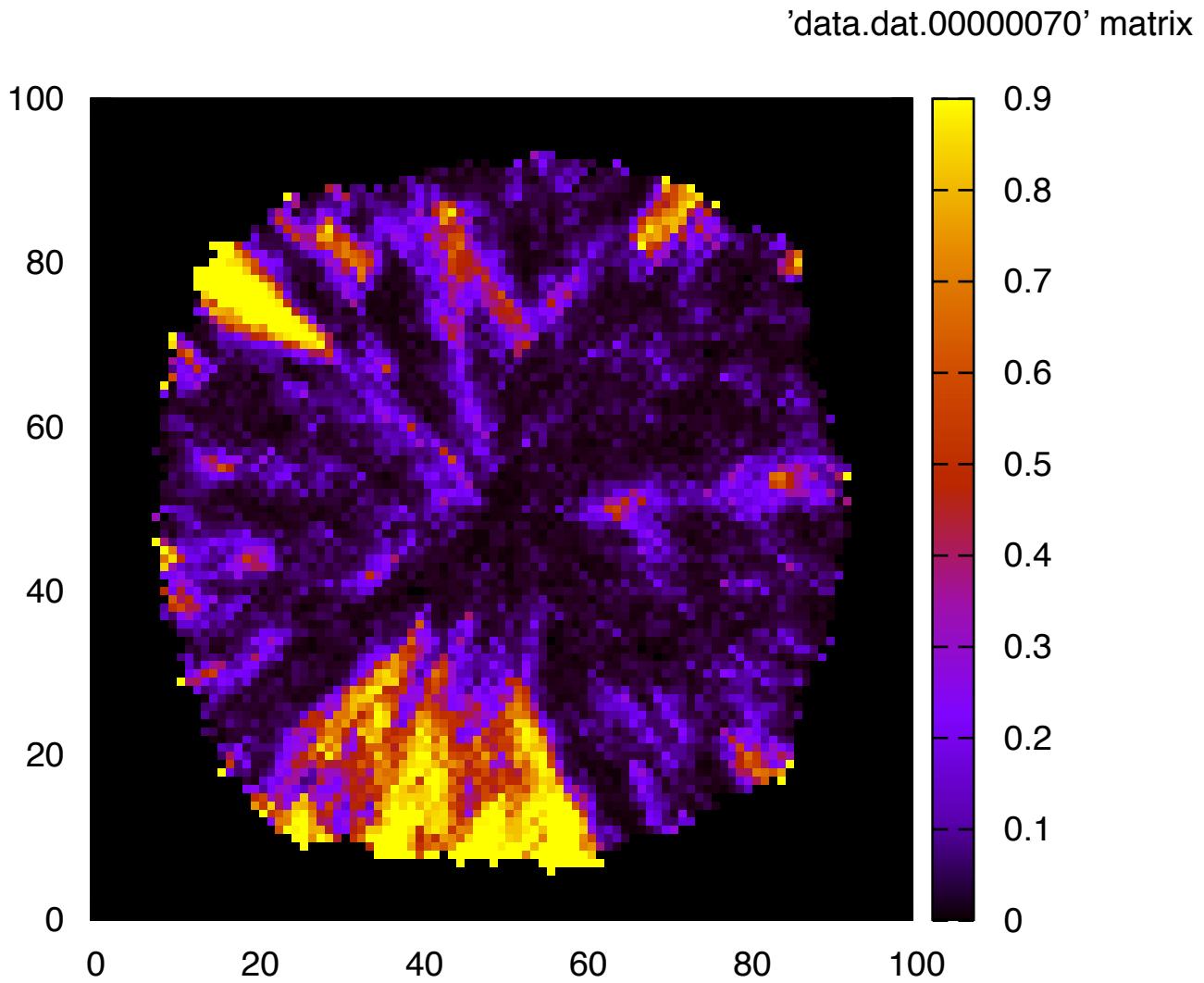
EX3: 2D - TRACE 1 (HIGH, F=1)



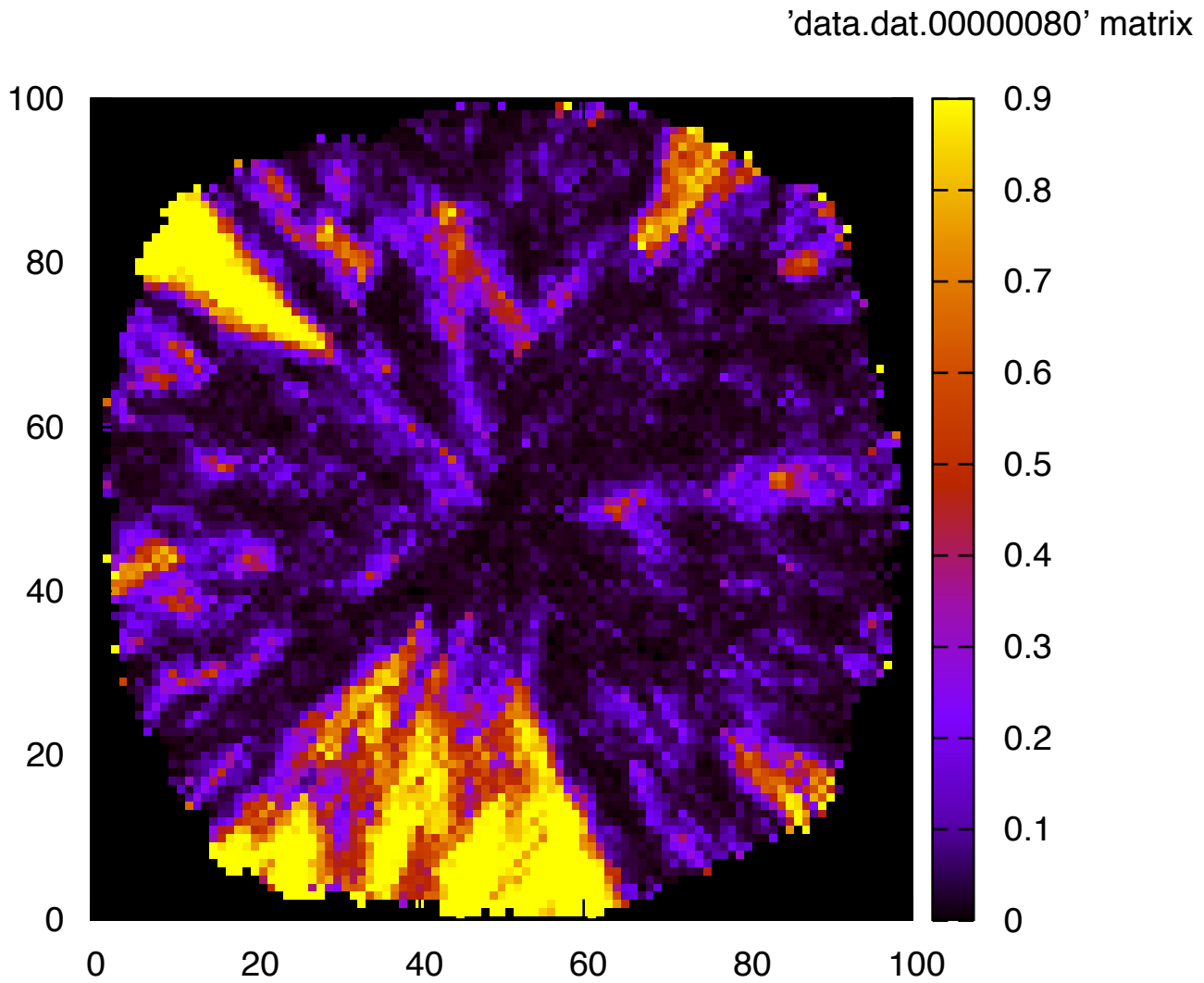
EX3: 2D - TRACE 1 (HIGH, F=1)



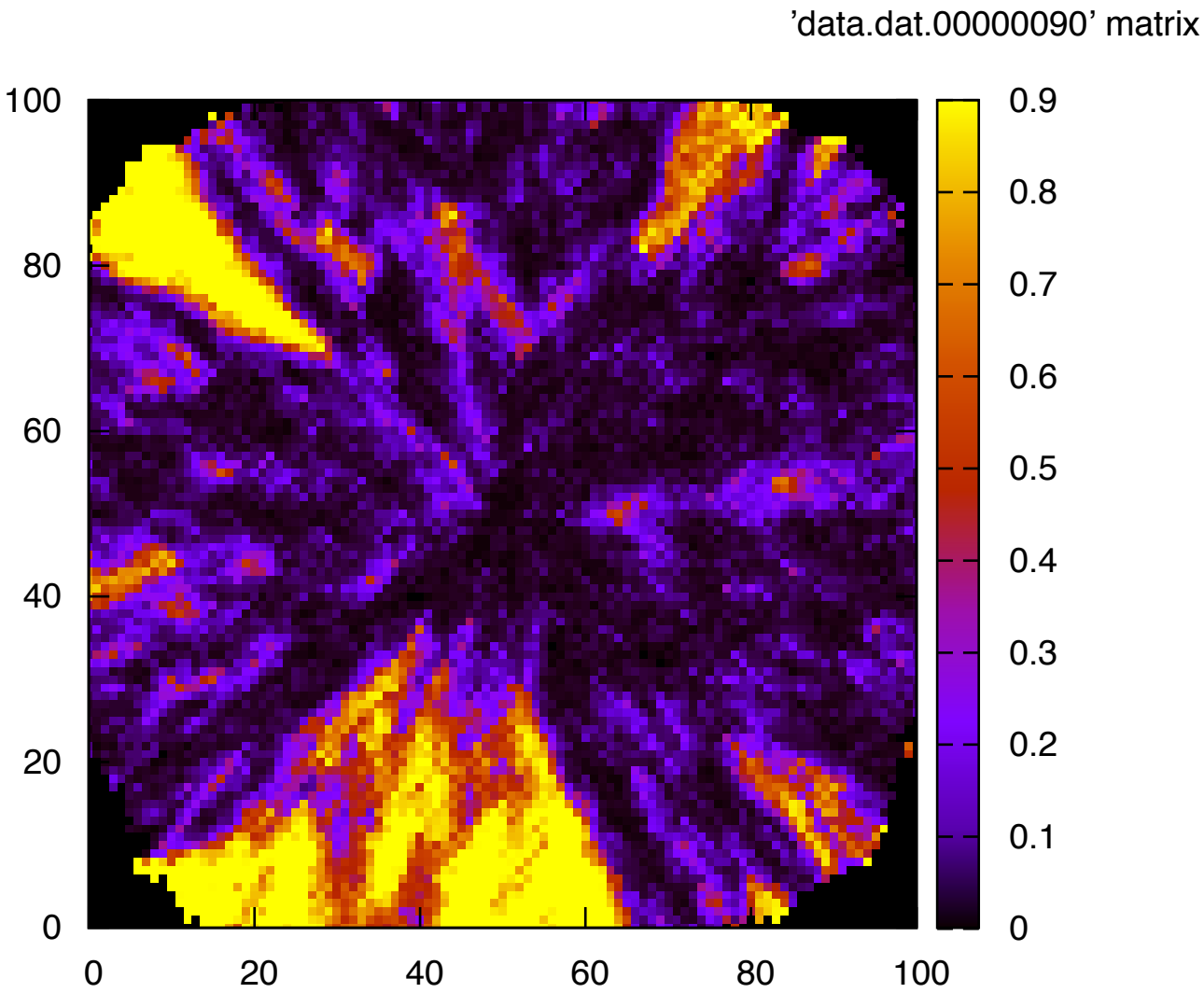
EX3: 2D - TRACE 1 (HIGH, F=1)



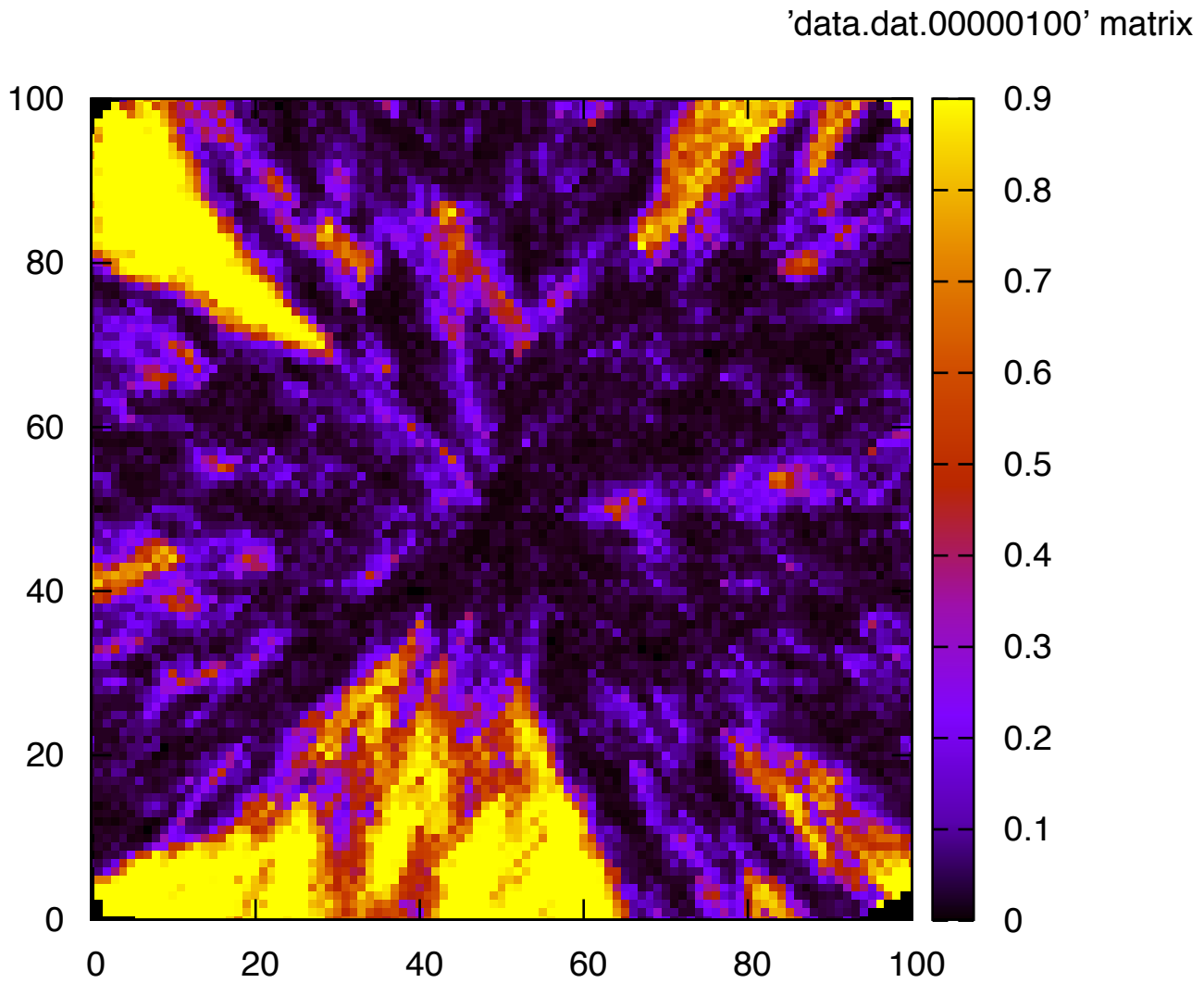
EX3: 2D - TRACE 1 (HIGH, F=1)



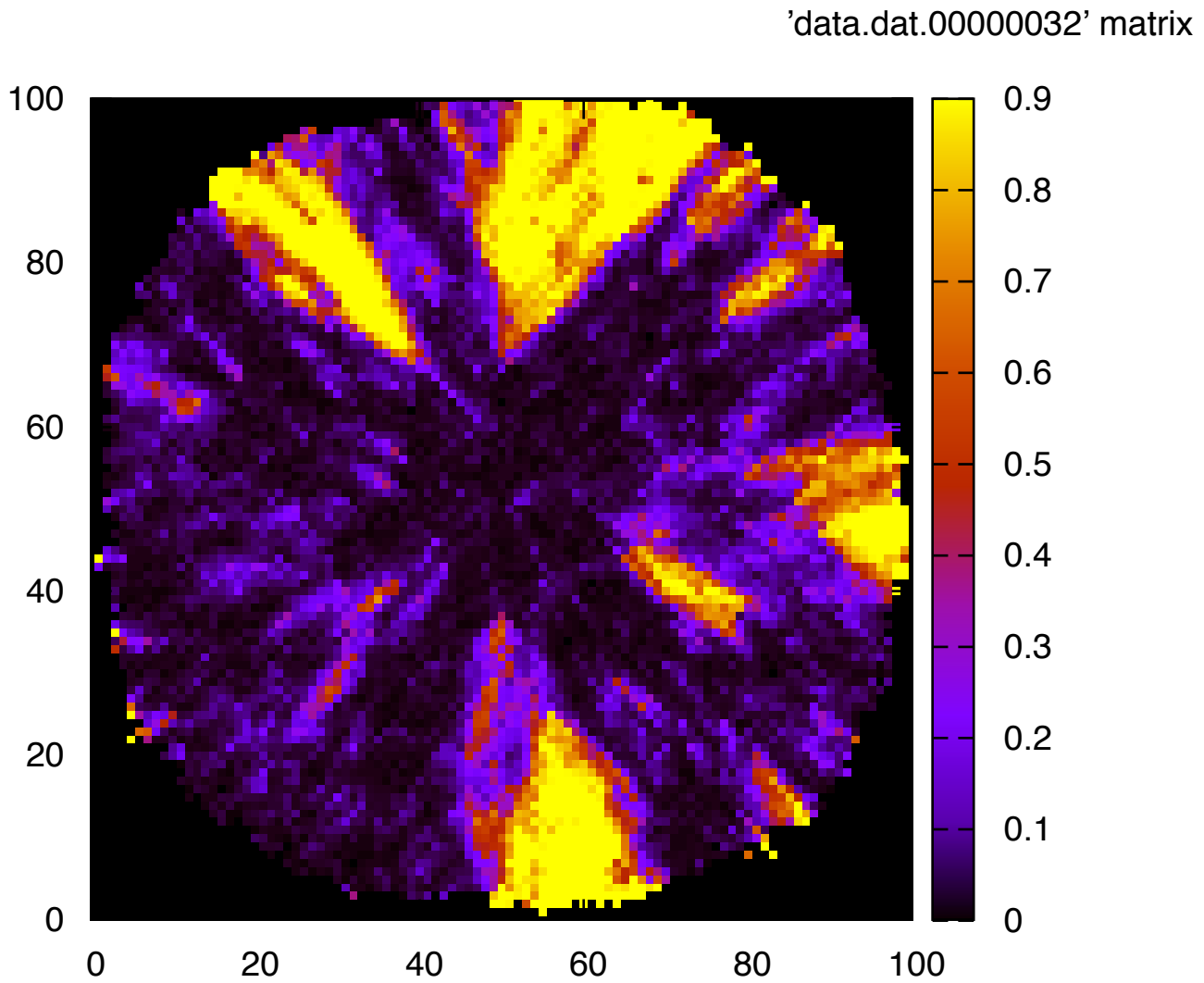
EX3: 2D - TRACE 1 (HIGH, F=1)



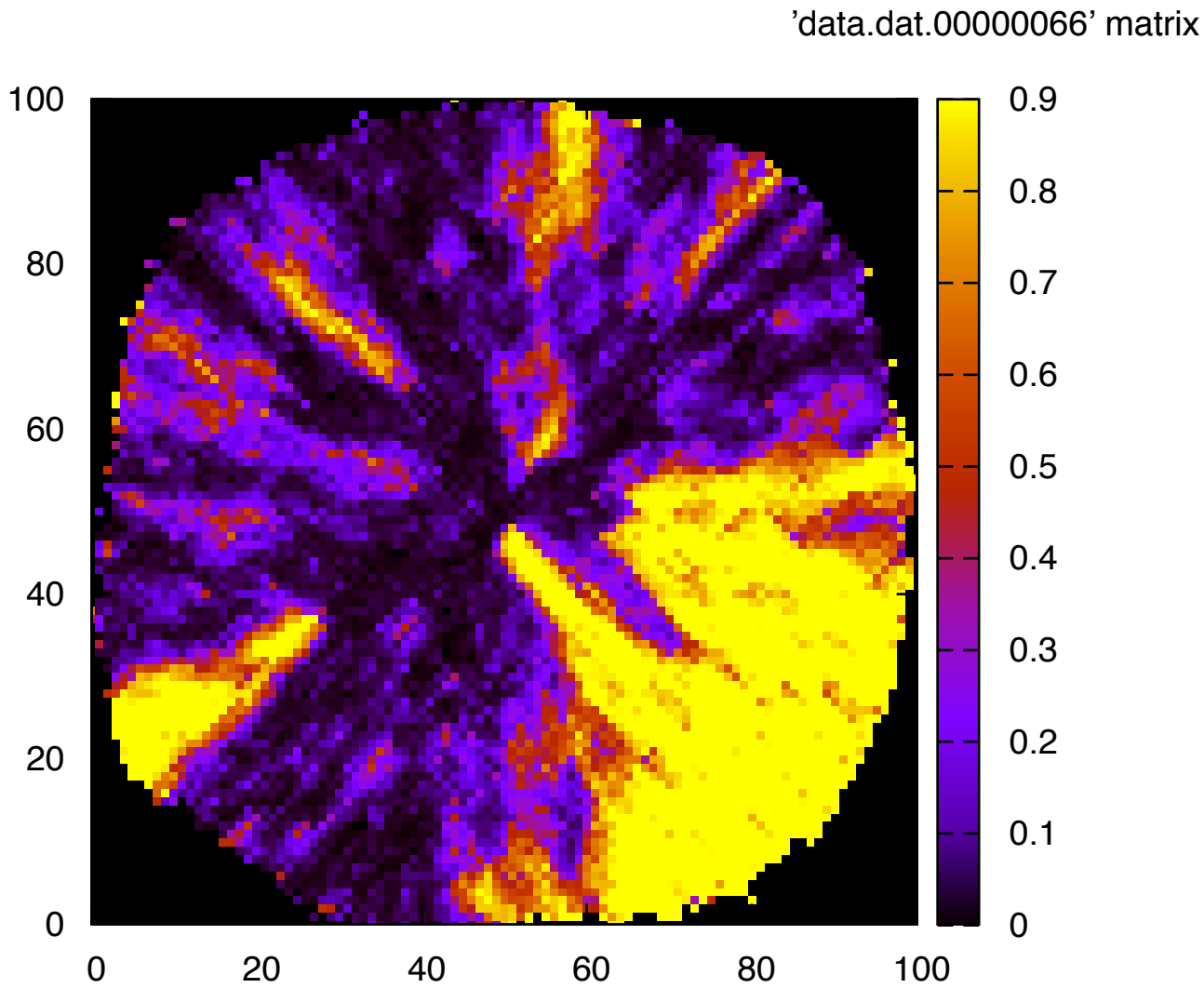
EX3: 2D - TRACE 1 (HIGH, F=1)



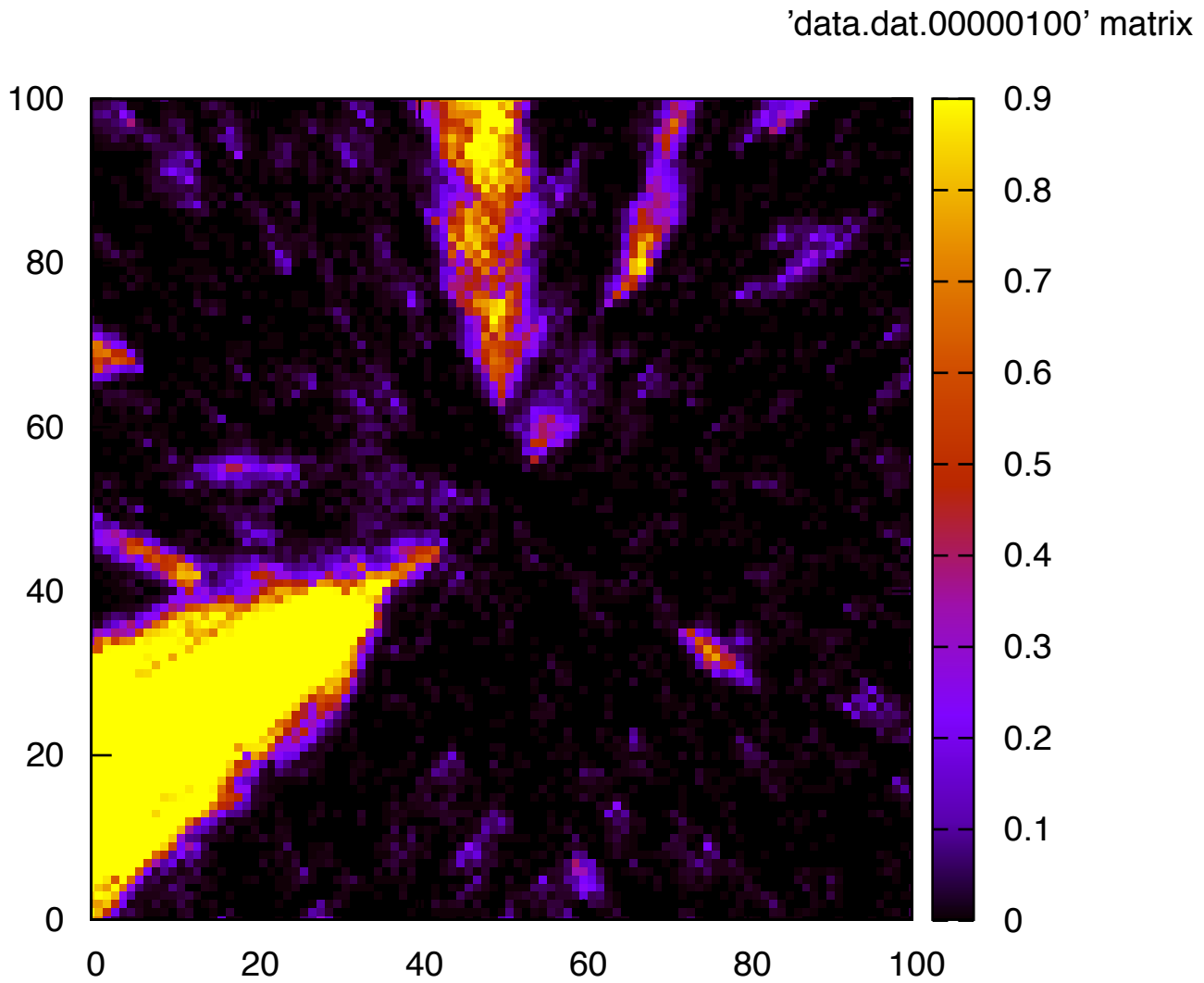
EX3: 2D - TRACE 2 (HIGH, F=1)



EX3: 2D - TRACE 3 (HIGH, F=1)

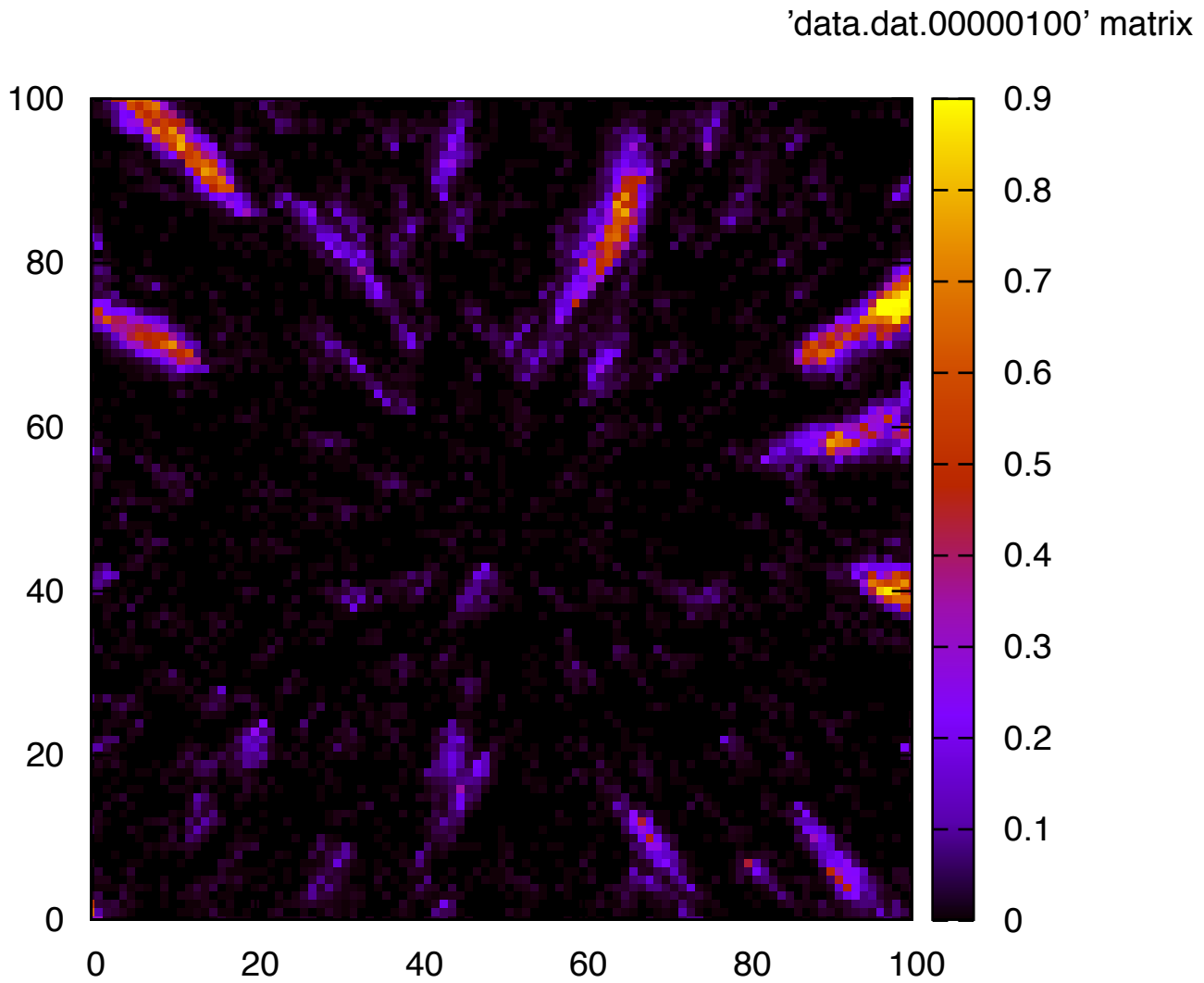


EX3: 2D - VARYING FITNESS, TRACE 1 (MEDIUM, F=1)



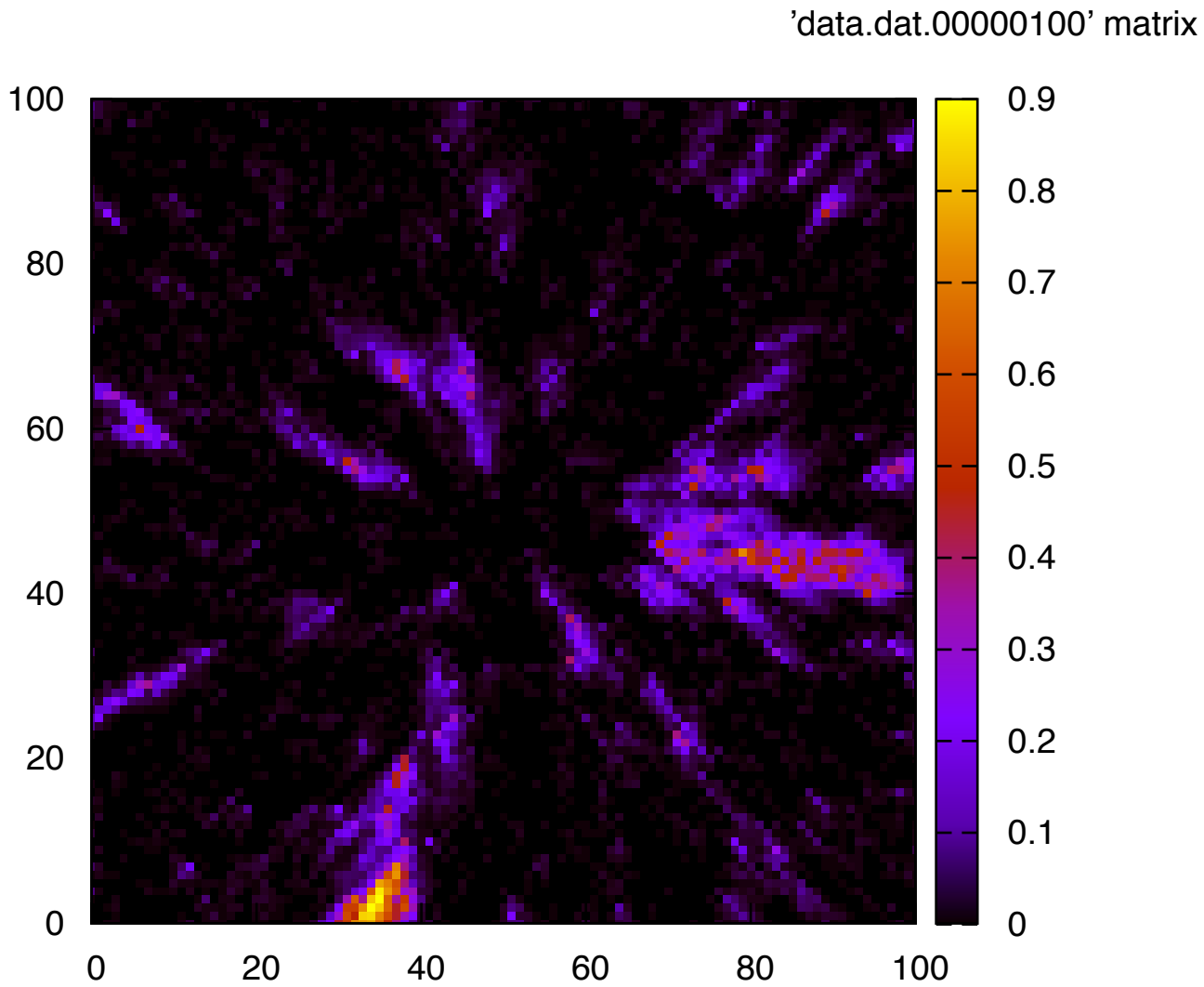
monika.heiner@b-tu.de

EX3: 2D - VARYING FITNESS, TRACE 2 (MEDIUM, F=1)

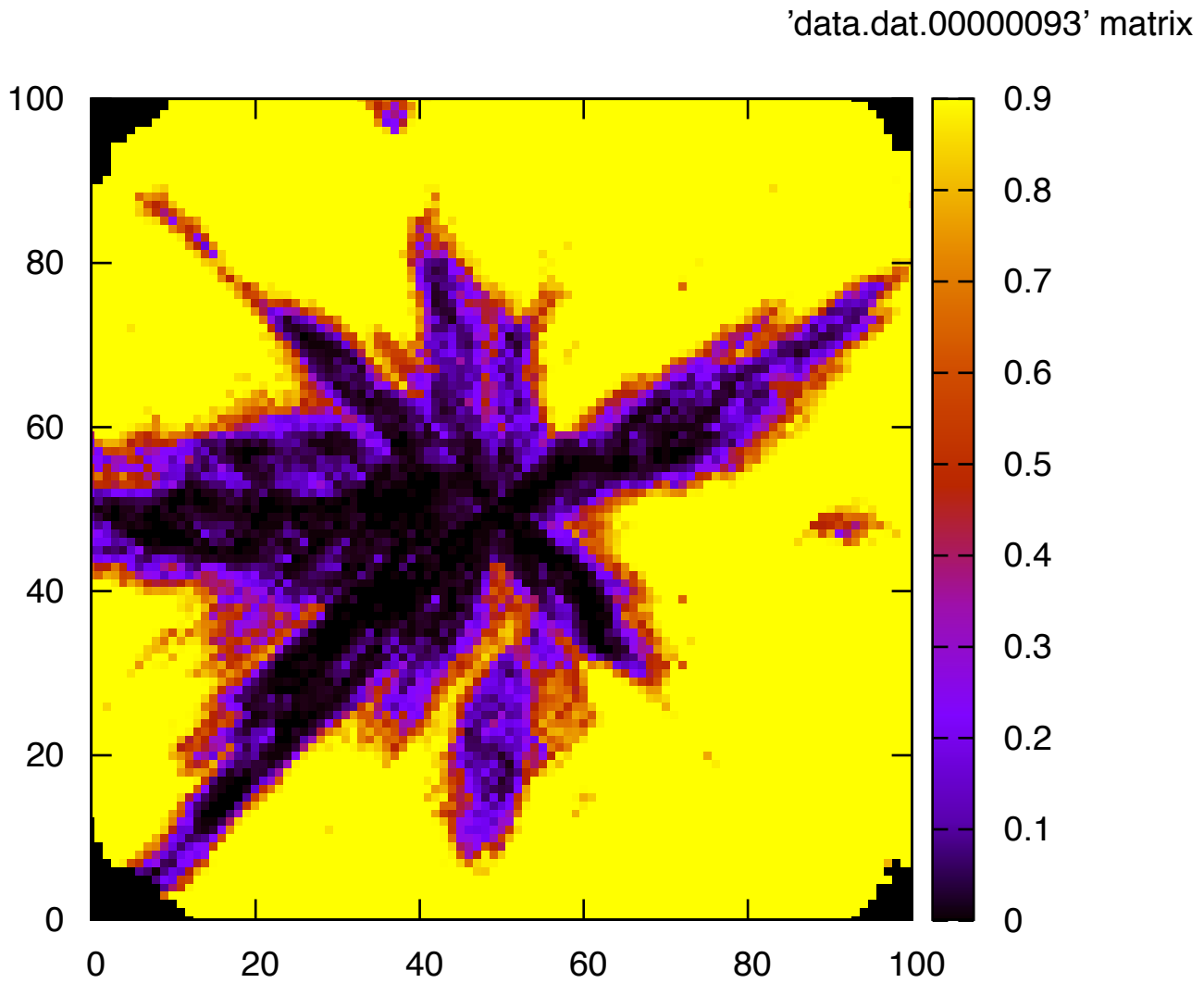


monika.heiner@b-tu.de

EX3: 2D - VARYING FITNESS, TRACE 1 (MEDIUM, F=0.99)

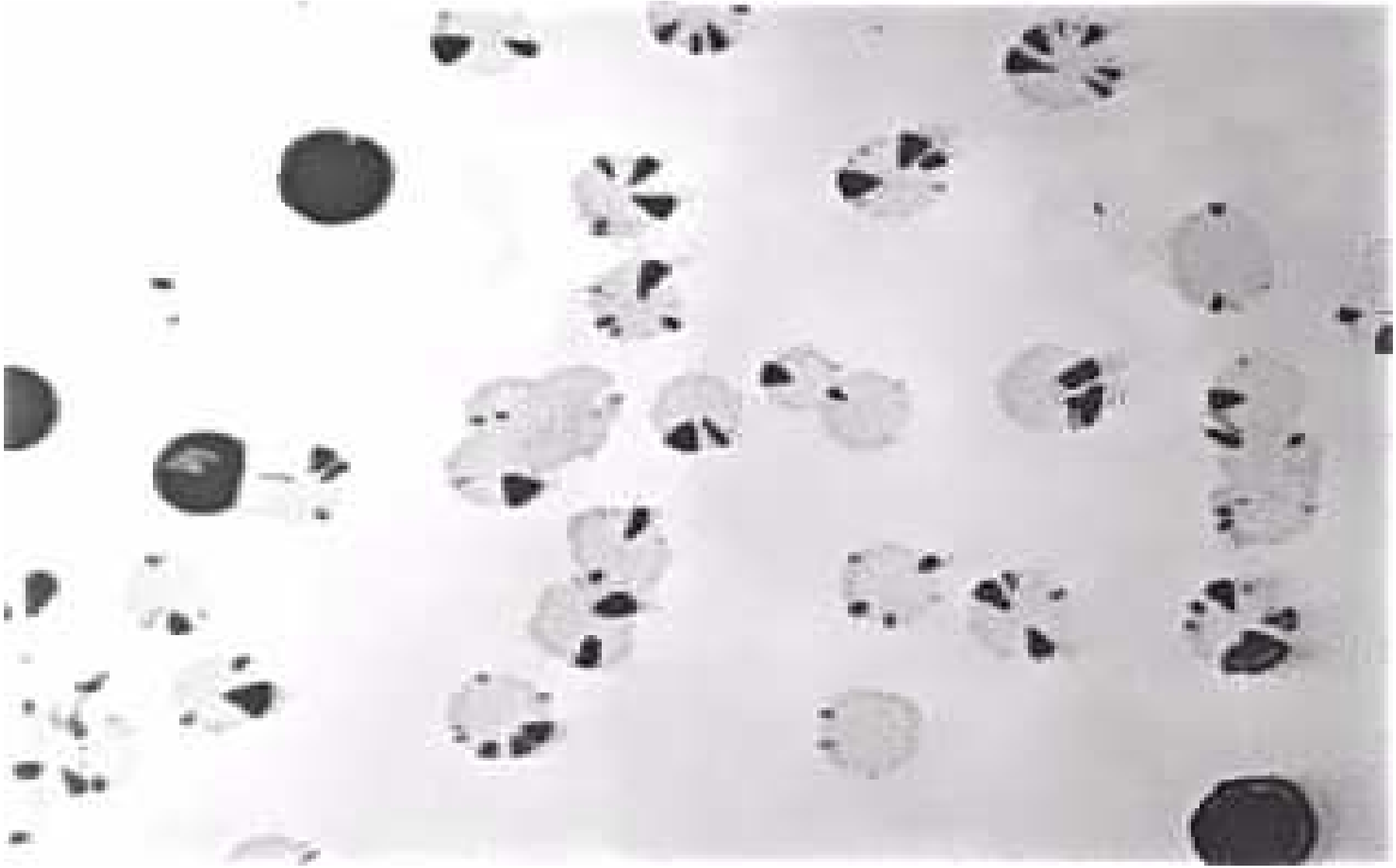


EX3: 2D - VARYING FITNESS, TRACE 1 (MEDIUM, F=0.90)

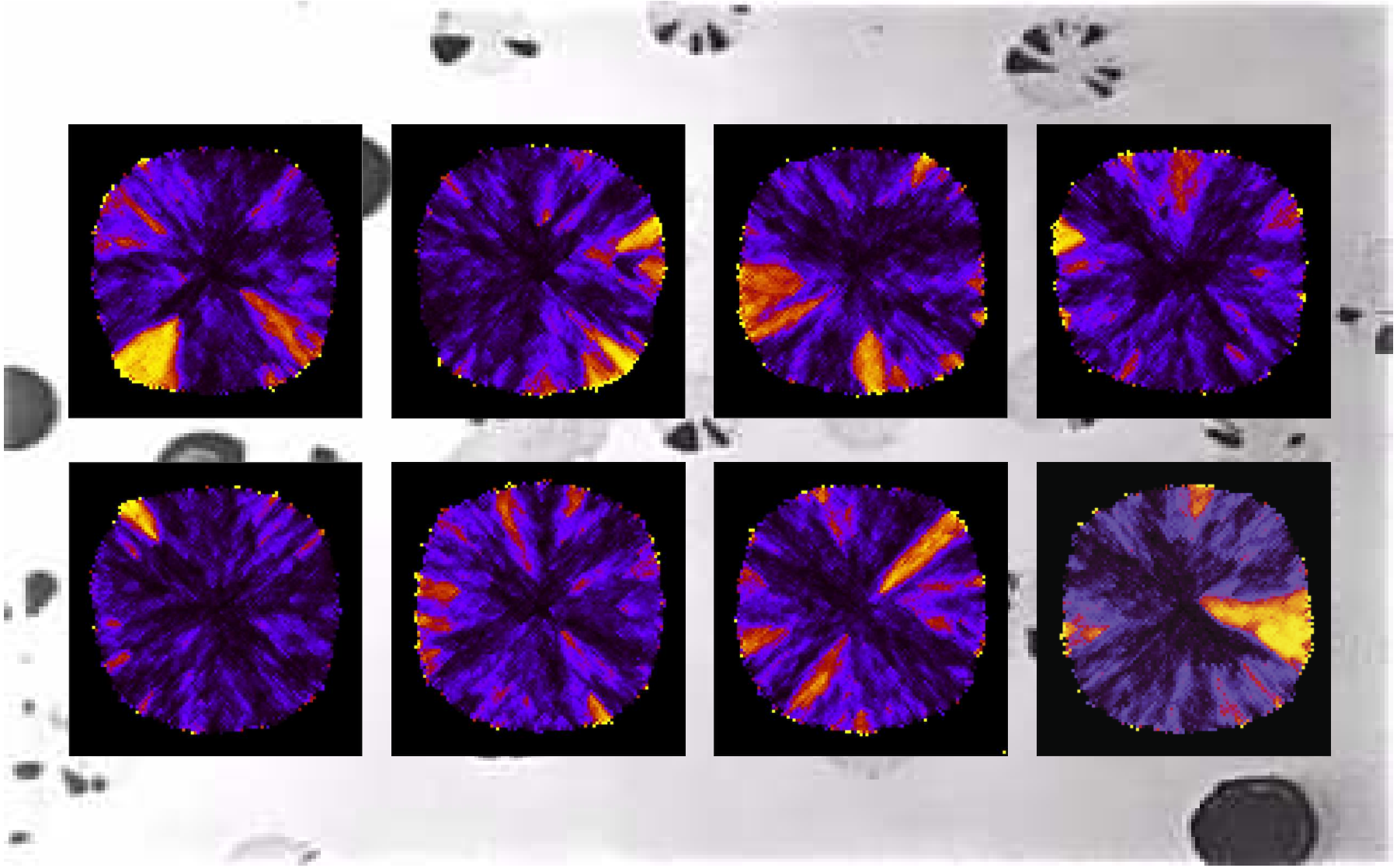


monika.heiner@b-tu.de

EX3: SOME FINAL STATES (HIGH, F=1)



EX3: SOME FINAL STATES (HIGH, F=1)



WHAT NEXT?

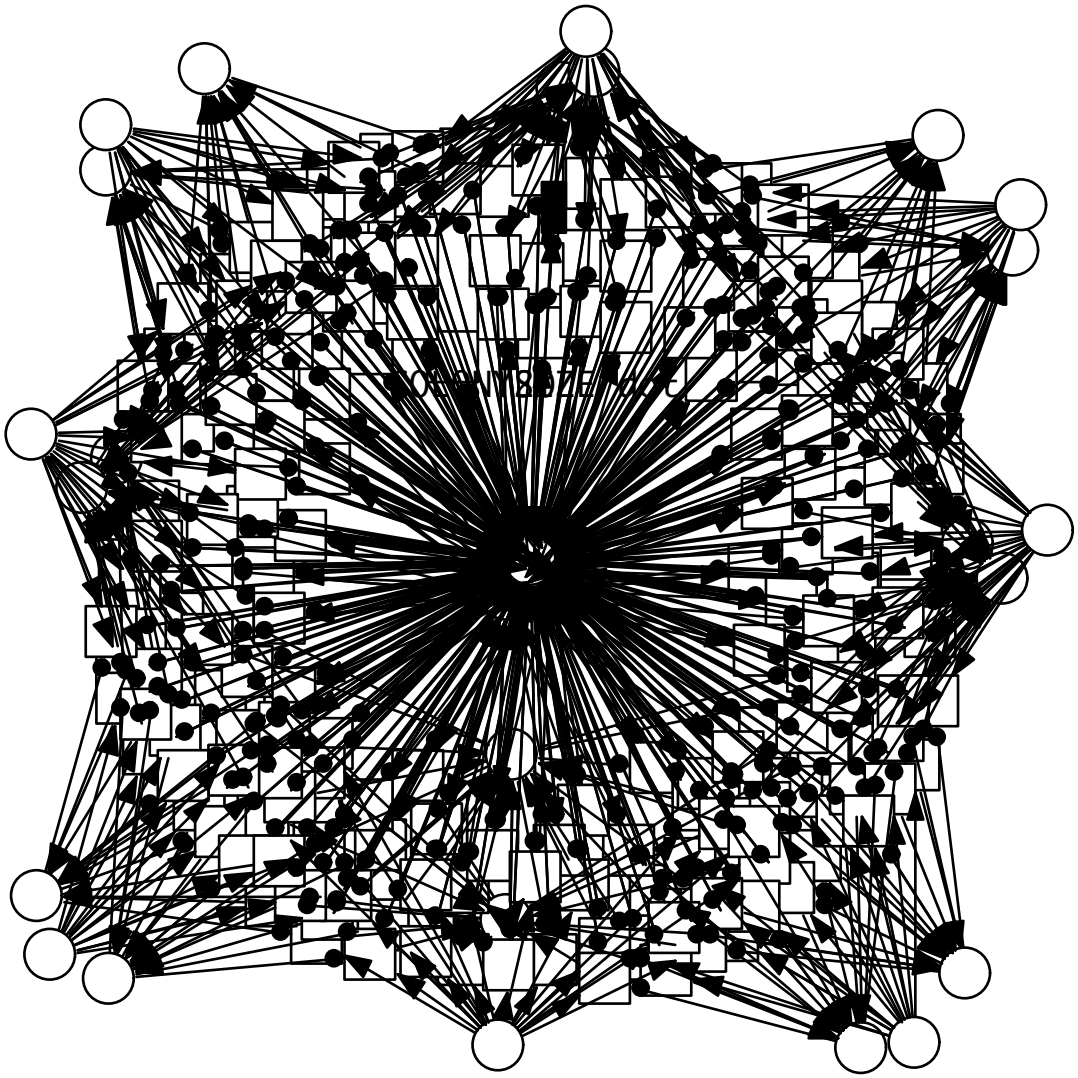
- **how to analyse visual data?** **-> CMSB 2013**
  - -> *auxiliary variables derived from model variables*
  - -> *clustering techniques*
  - -> *shape recognition*
  - -> *visual analytics*

- **use model to predict**
  - -> *mutation rates by measuring the mutation sectors,*

    . . . or just the number of sectors?
  - -> *fitness by measuring angel of sectors*

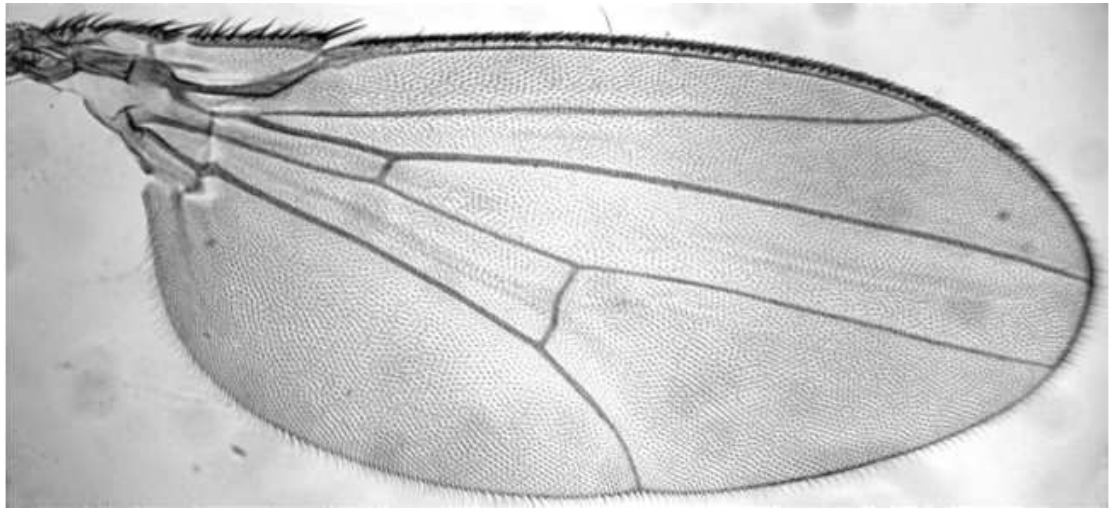
- **possible model extensions / variations**
  - -> *fine tuning of biofilm thickness*
  - -> *multiple gene on/off and their dependencies*
  - -> *log pedigree and/or mobility*

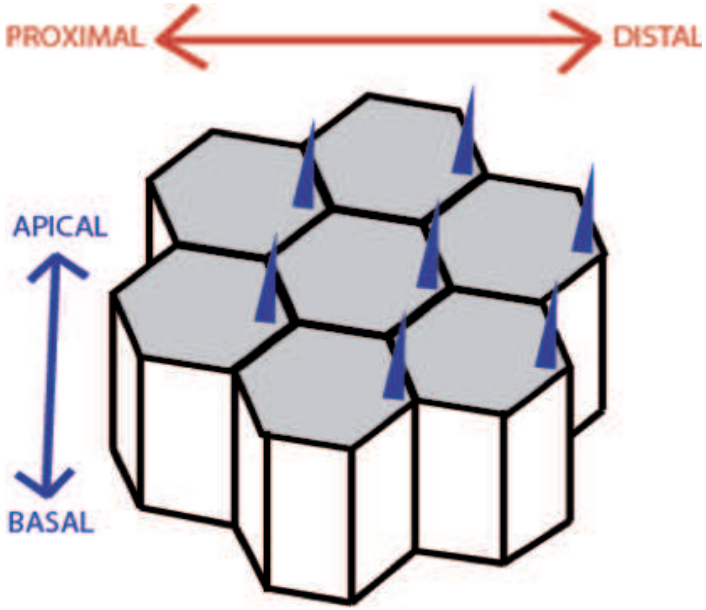
PHASE VARIATION, PLAIN MODEL (3X3)



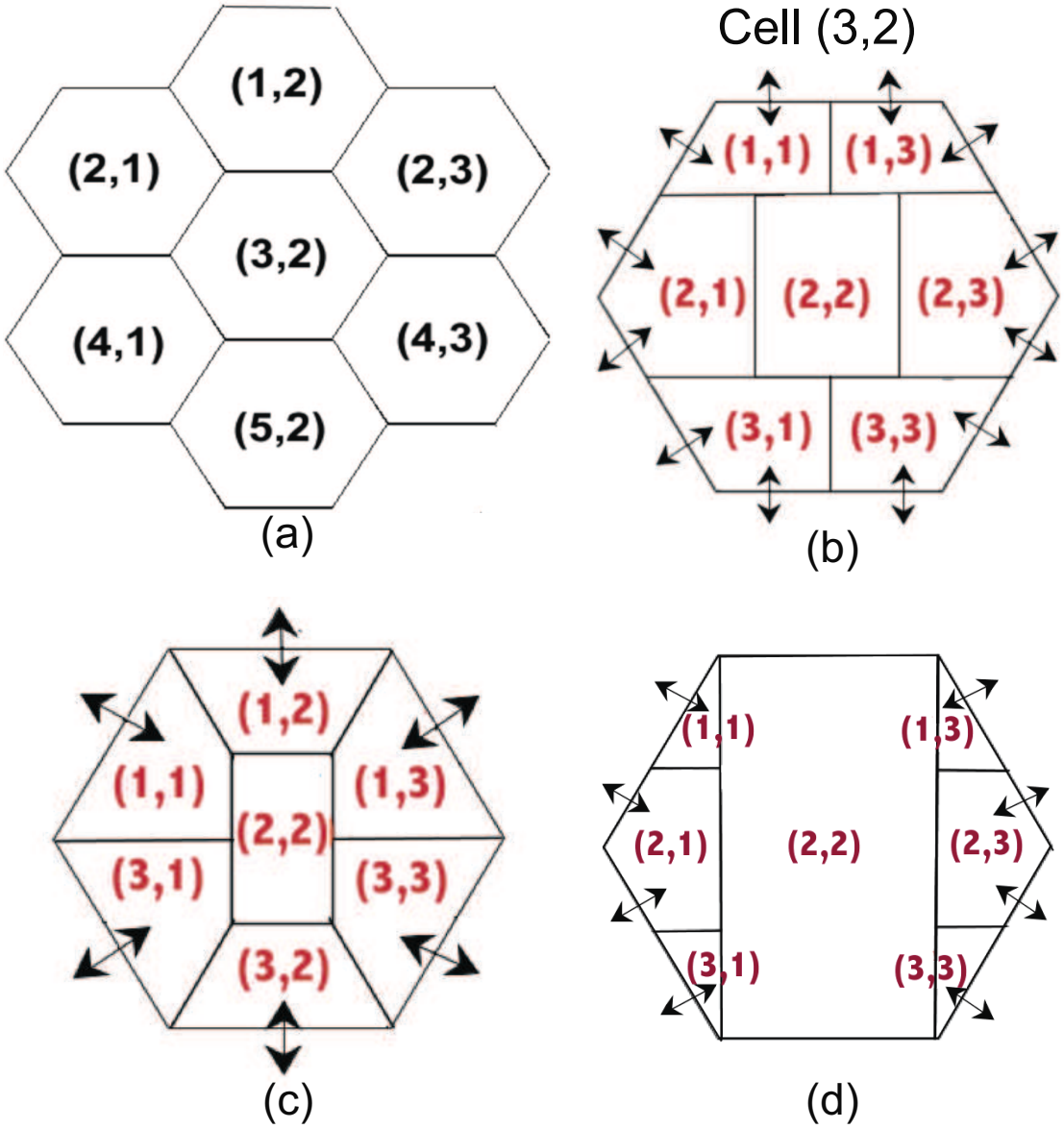
EXAMPLE 4:

PLANAR CELL POLARITY IN FLY WING

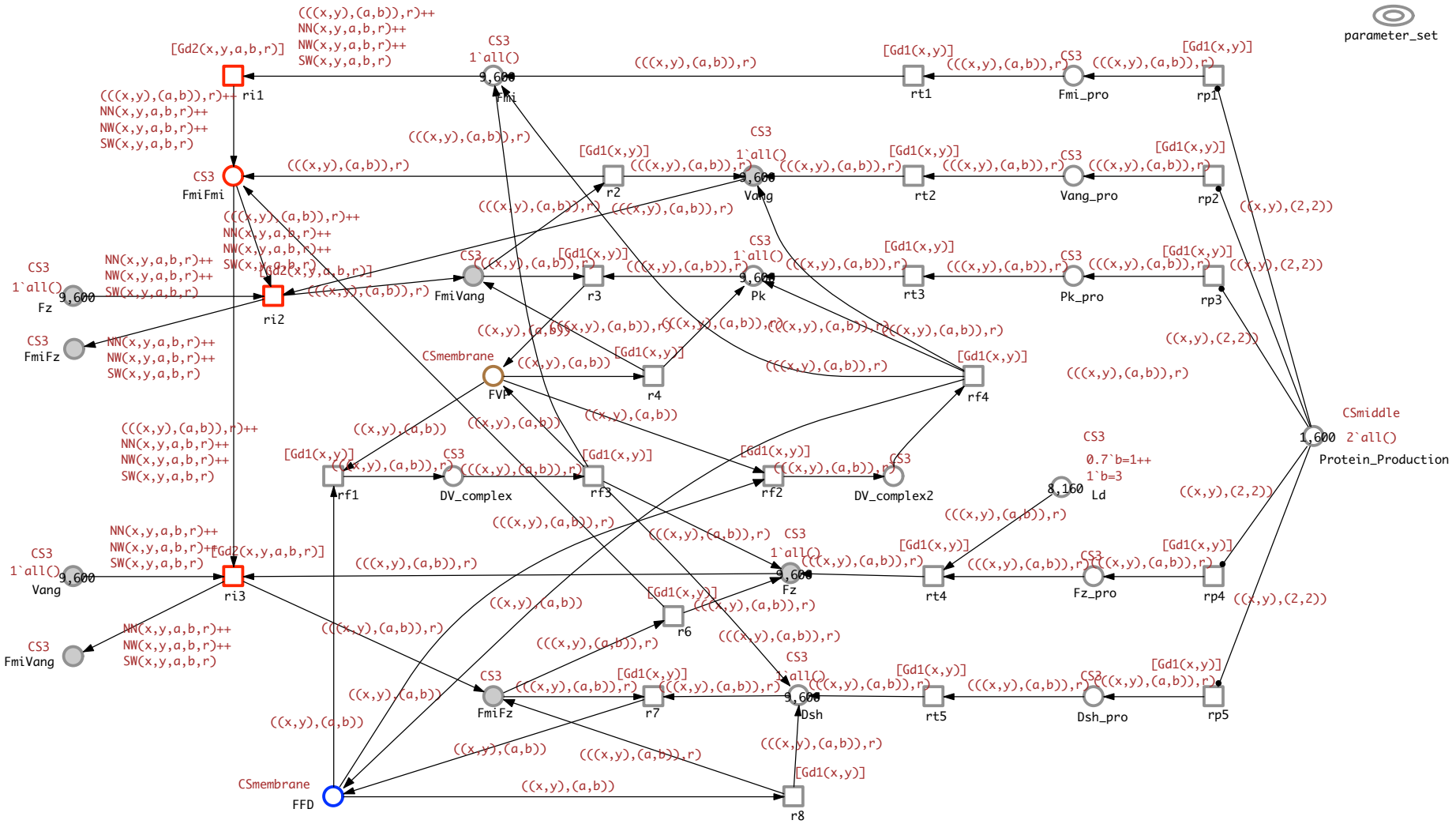




[BIOPPN 2011]
[CMSB 2011]

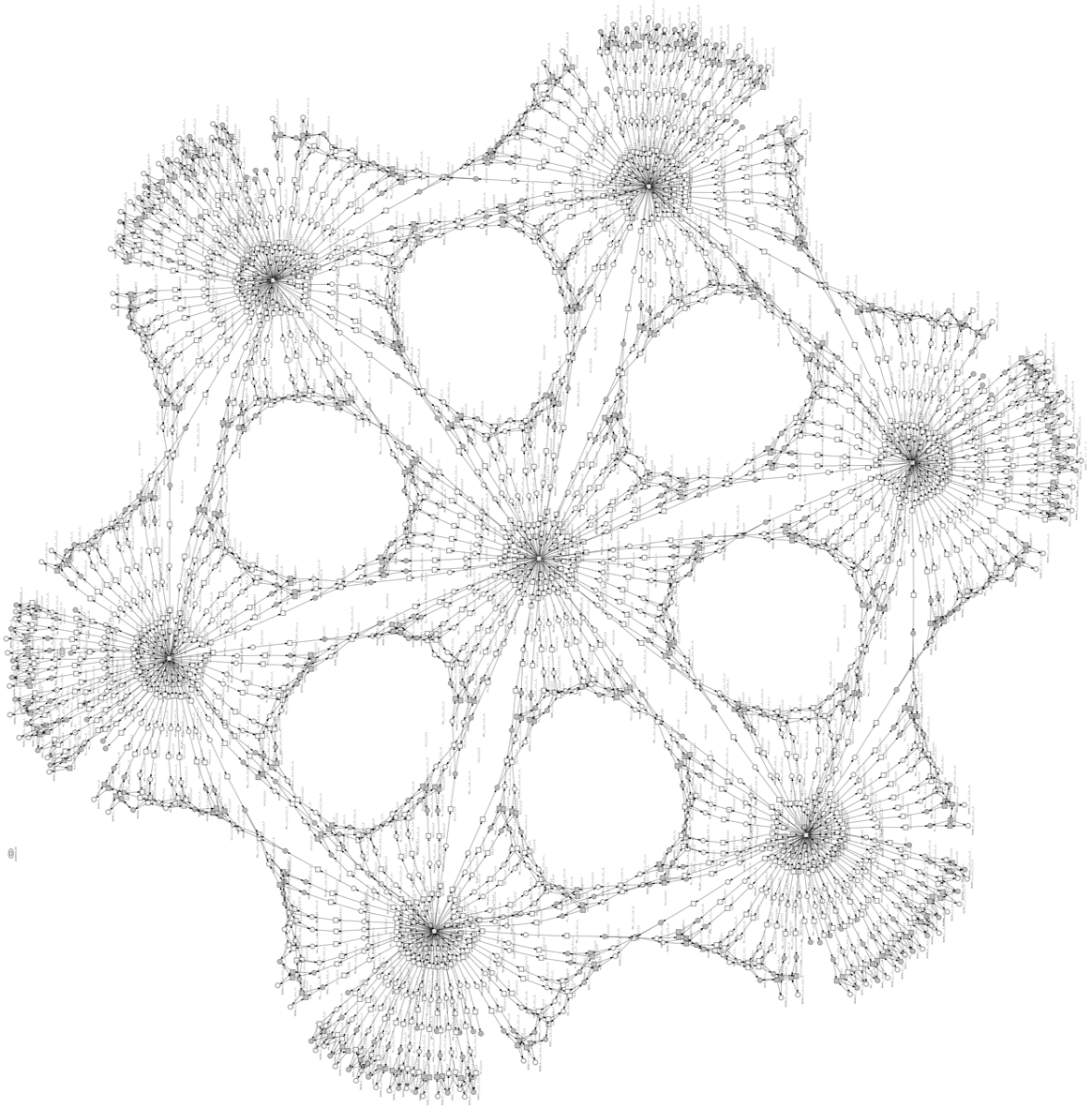

(1,2)
(2,1)
(2,3)
(3,2)
(4,1)
(4,3)
(5,2)
(a)
Cell (3,2)
(1,1)
(1,3)
(2,1)
(2,2)
(2,3)
(3,1)
(3,3)
(b)
(1,2)
(1,1)
(1,3)
(2,2)
(3,1)
(3,3)
(3,2)
(c)
(1,1)
(1,3)
(2,1)
(2,2)
(2,3)
(3,1)
(3,3)
(d)

EX4: PLANAR CELL POLARITY

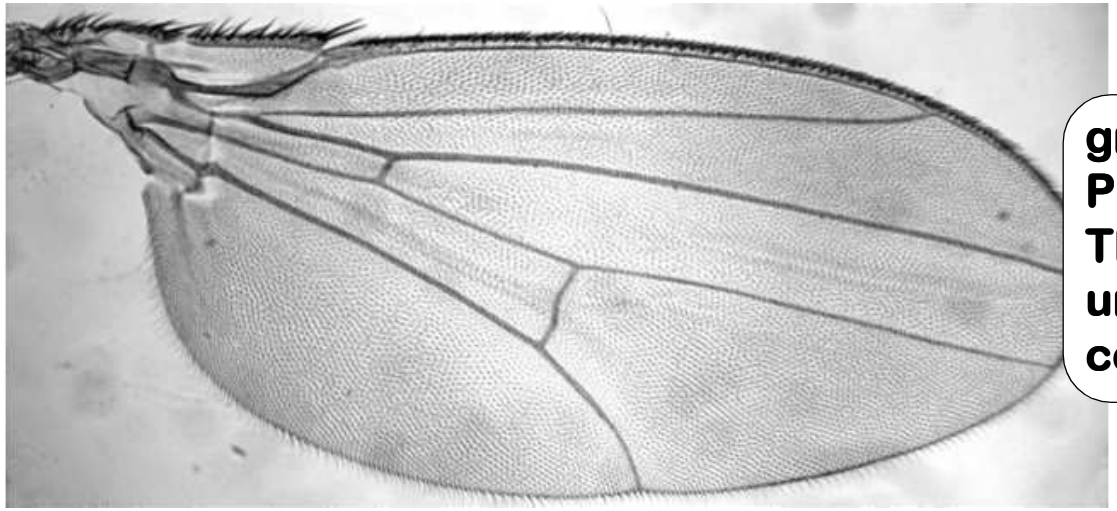


[QIAN GAO, PHD THESIS 2013]

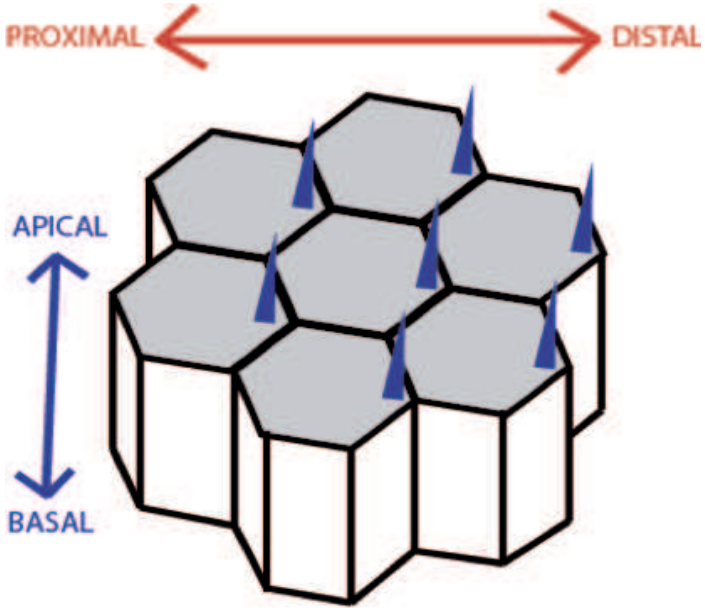
EX4: PLANAR CELL POLARITY, PLAIN MODEL (7 CELLS)



monika.heiner@b-tu.de



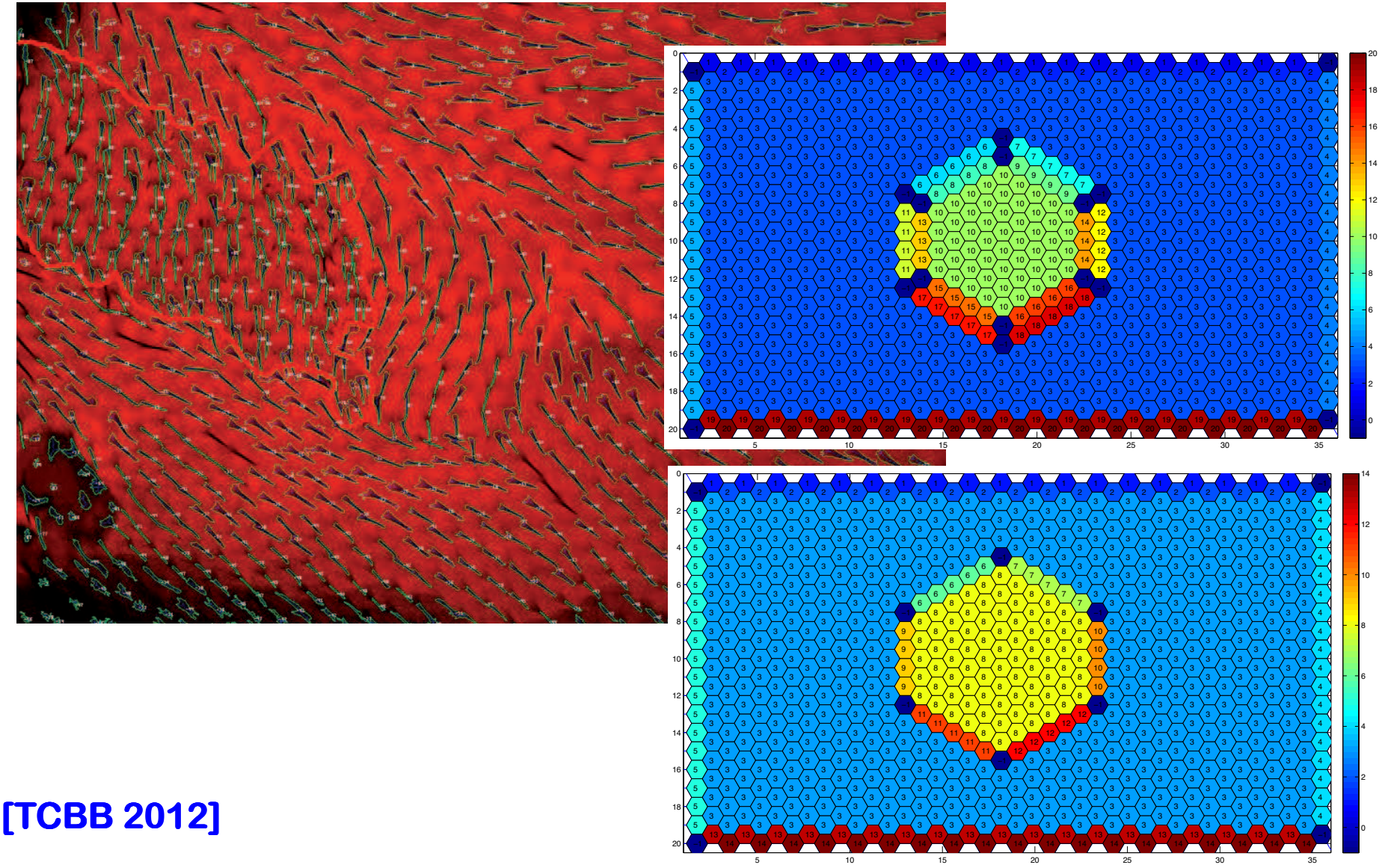
grid size:	**40 x 40**
PLACES:	**164,000**
TRANSITIONS:	**229,686**
unfolding:	**4 min**
cont. simulation:	**2 h**



[BIOPPN 2011]
[CMSB 2011]

EX4 - PLANAR CELL POLARITY

EX4 - PLANAR CELL POLARITY



[TCBB 2012]

monika.heiner@b-tu.de

CREDITS

- **EPSRC Research Grant EP/I036168/1**

- **collaborators**

  David Gilbert, Brunel University, London, UK

  Wolfgang Marwan, Otto-von-Guericke University, Magdeburg, Germany

- **case studies**

  Turing Patterns - Mary Ann Blätke, Fei Liu

  bacterial colony - Ovidiu Parvu, David Gilbert, Nigel Saunders

  PCP in fly wing - Pam Gao, David Gilbert, David Tree

- **Snoopy + Charlie + Marcie development**

  Christian Rohr, Fei Liu, Mostafa Herayj

  Martin Schwarick, Jan Wegener

- **plots**

  Mary Ann Blätke, Daniele Maccagnola, Ovidiu Parvu, Christian Rohr, Jan Wegener

ADVANTAGES OF THIS APPROACH

- **the spatial modelling principle can be equally applied to all paradigms**
  -> *qualitative, stochastic, continuous, and hybrid*
  -> *model transformations preserve all spatial attributes*

- **all space-related information is encoded in colour**
  -> *reuse in other models*

- **changing the notion of space**
  -> *adapt colour-related definitions*
  -> *net structure itself needs not to be touched.*

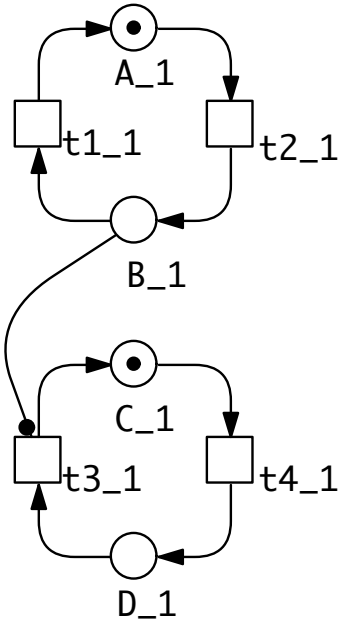
- **use of a priori finitely discretised space preserves model analysibility**

- **automatic unfolding**
  -> *reuse of all analysis and simulation techniques of uncoloured Petri nets*

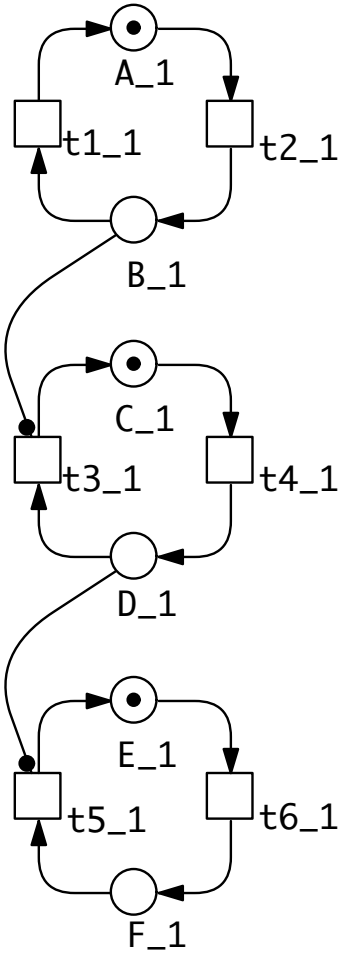
How to Encode Space, Version 1 - the Big Cons

monika.heiner@b-tu.de

CONS

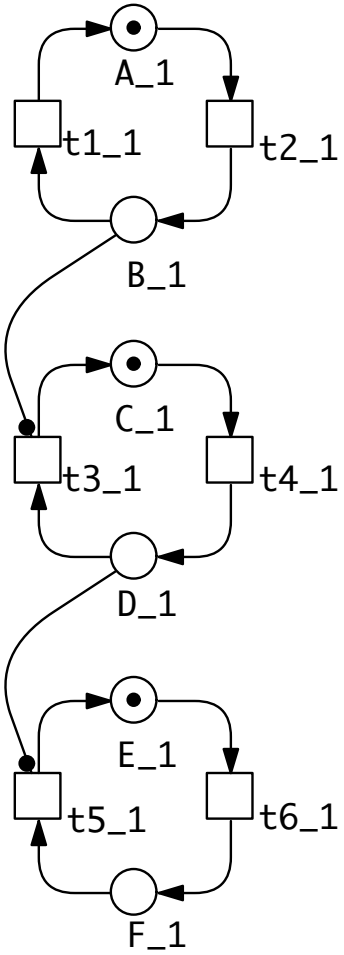


CONS

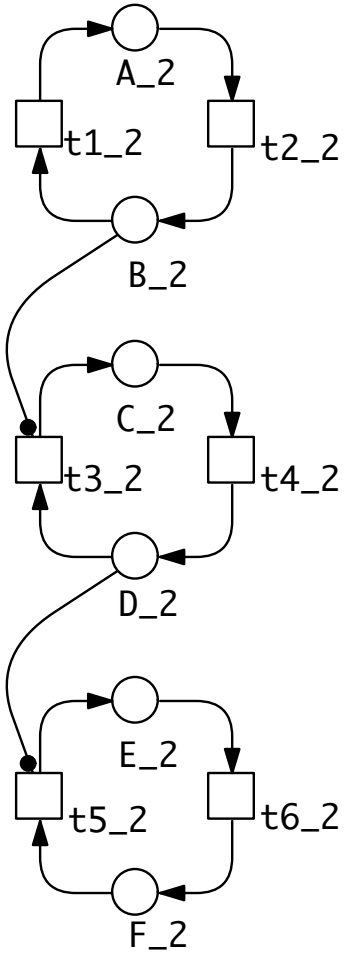


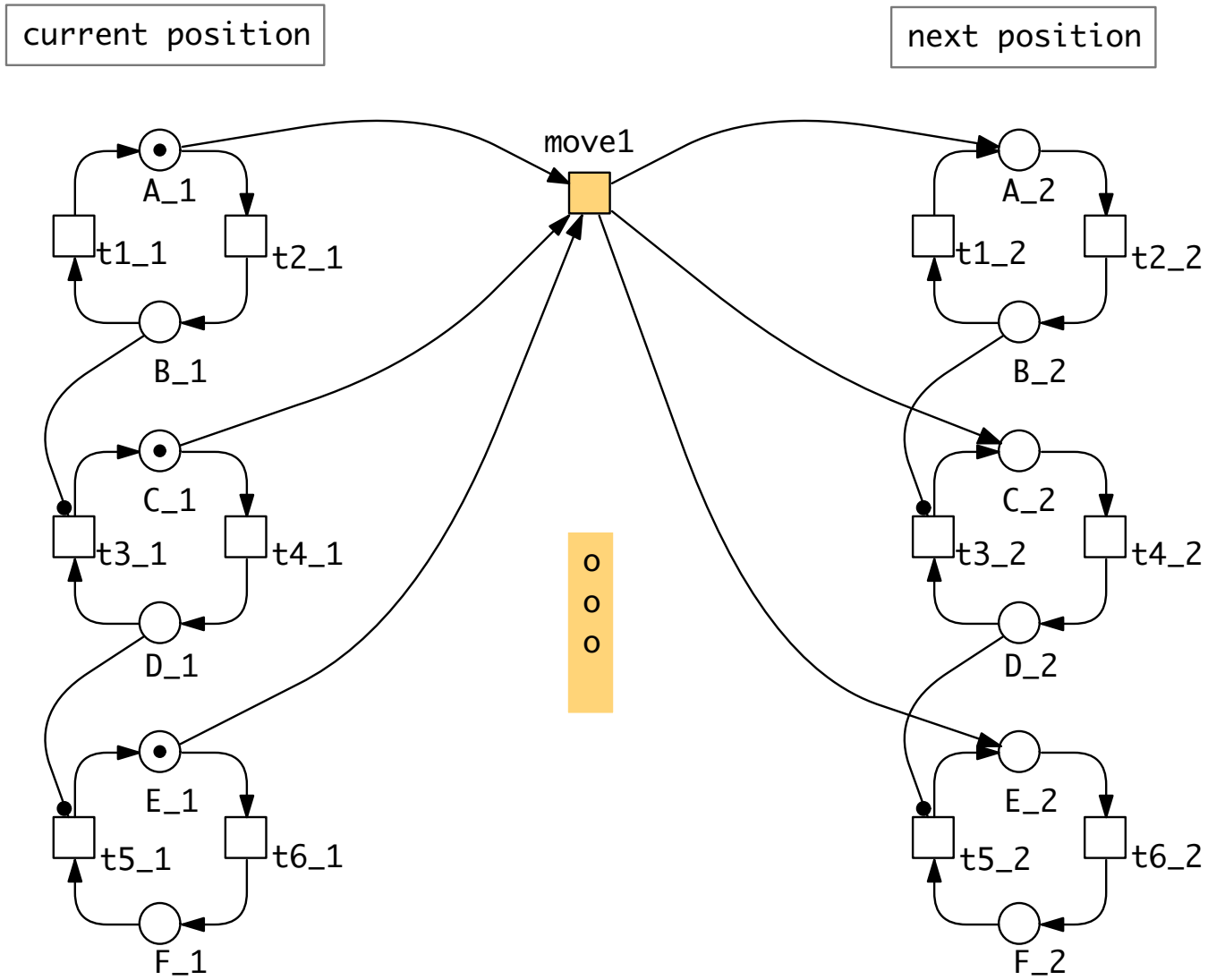
CONS

current position

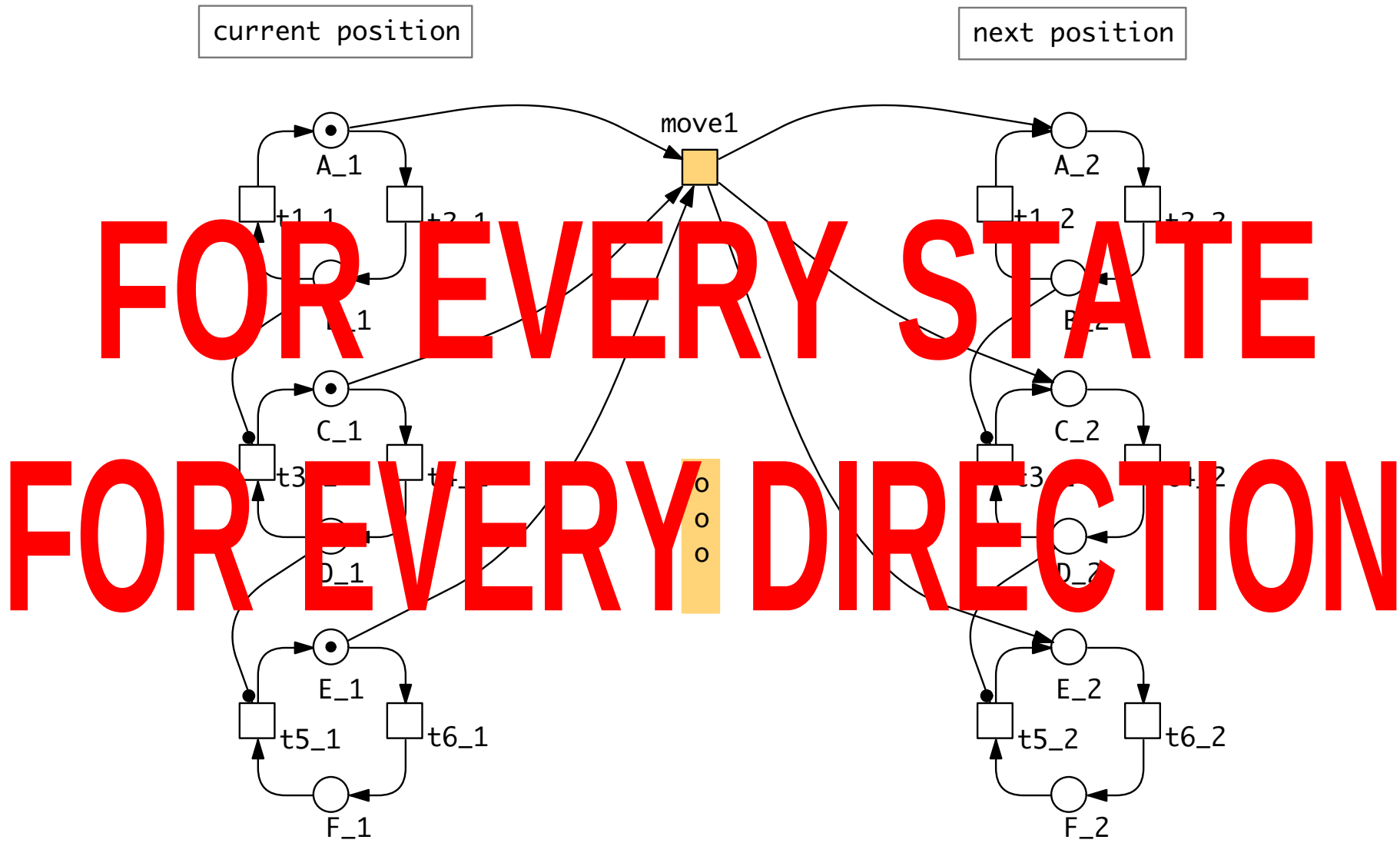


next position




current position
next position
move1
A_1
t1_1
t2_1
B_1
C_1
t3_1
t4_1
D_1
E_1
t5_1
t6_1
F_1
A_2
t1_2
t2_2
B_2
C_2
t3_2
t4_2
D_2
E_2
t5_2
t6_2
F_2
o
o
o

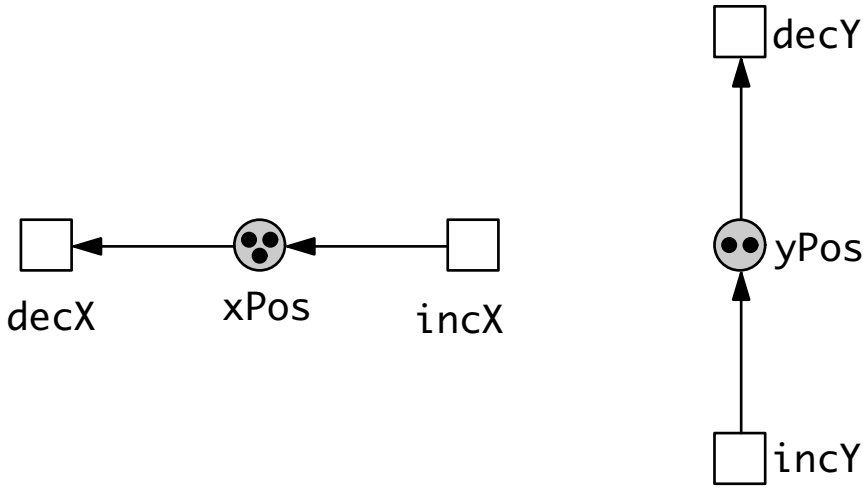
CONS


current position
next position
move1
A_1
A_2
t1_1
t2_1
t1_2
t2_2
B_1
B_2
C_1
C_2
t3_1
t4_1
t3_2
t4_2
D_1
D_2
E_1
E_2
t5_1
t6_1
t5_2
t6_2
F_1
F_2
FOR EVERY STATE
FOR EVERY DIRECTION

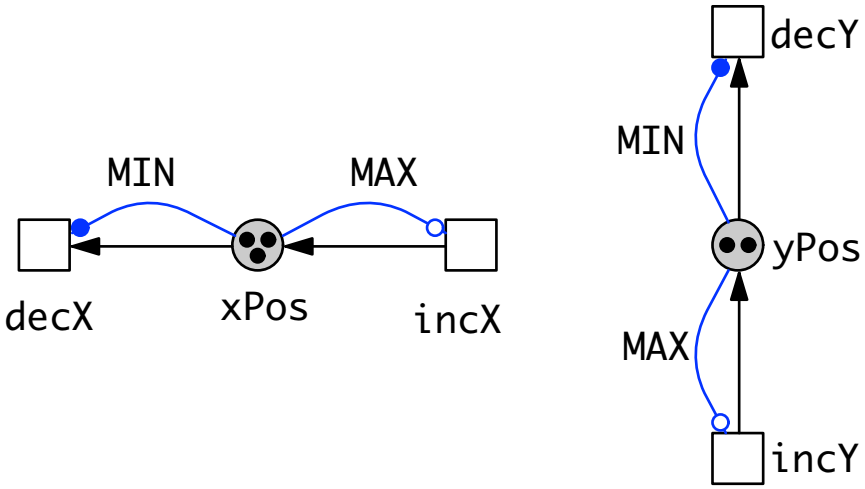
How to Encode Space ?
Version 2

ENCODING SPACE, VERSION 2

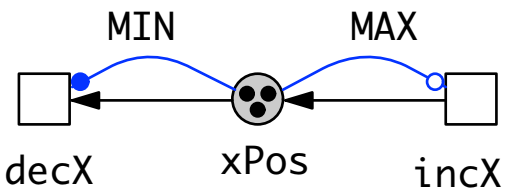
ENCODING SPACE, VERSION 2

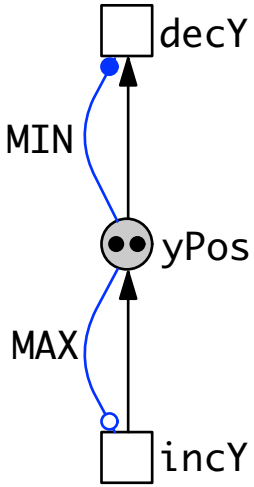


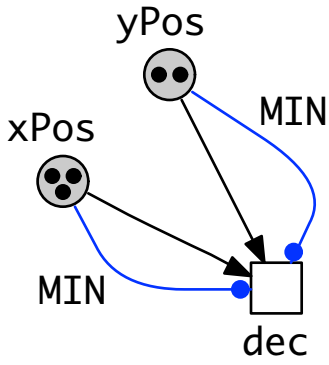
fence in

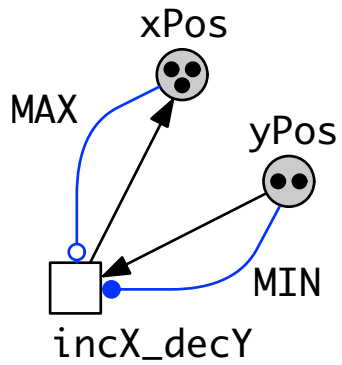


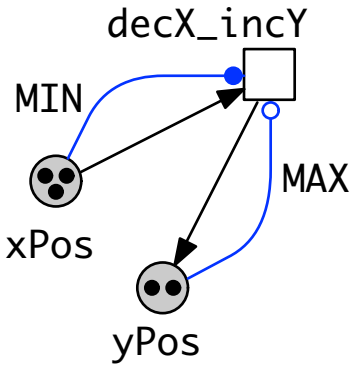
fence in

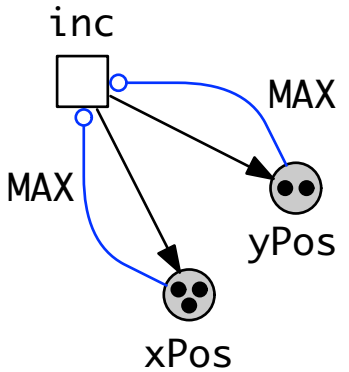

MIN
MAX
decX
xPos
incX


decY
MIN
yPos
MAX
incY


yPos
xPos
MIN
MIN
dec

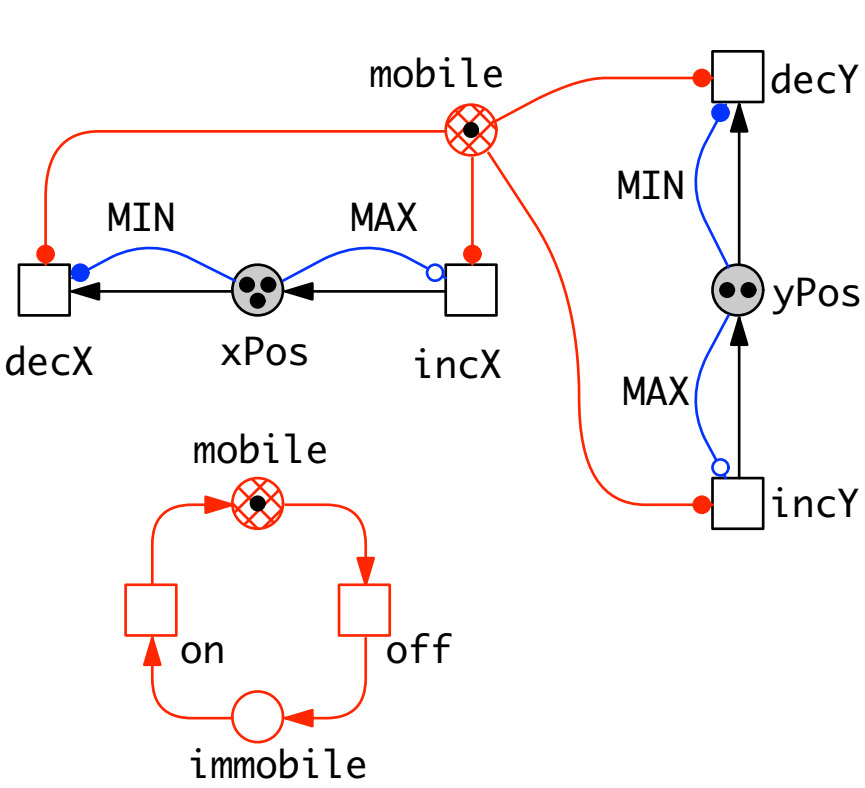

xPos
MAX
yPos
MIN
incX_decY

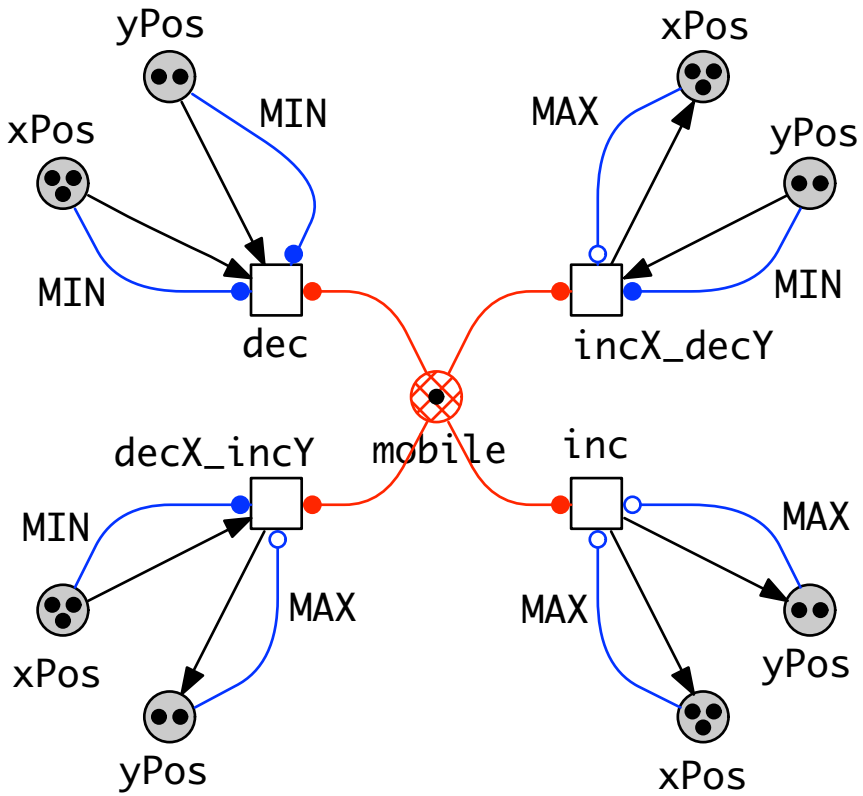

decX_incY
MIN
MAX
xPos
yPos


inc
MAX
MAX
yPos
xPos

ENCODING SPACE, VERSION 2

fence in





permit movement on/off

ENCODING SPACE, COMPARISON

VERSION1

- a priori finite space
- unfolding generates PN for the whole finite universe

  -> many places might be empty
- requires atomic moving objects

VERSION2

- potentially infinite space
- size of the unfolded PN does not dependent on size of the universe
- local states in moving objects possible

ENCODING SPACE, COMPARISON

VERSION1

- a priori finite space
- unfolding generates PN for the whole finite universe
  -> many places might be empty
- requires atomic moving objects

VERSION2

- potentially infinite space
- size of the unfolded PN does not dependent on size of the universe
- local states in moving objects possible

PRO

- state-dependent rates as usual

CON

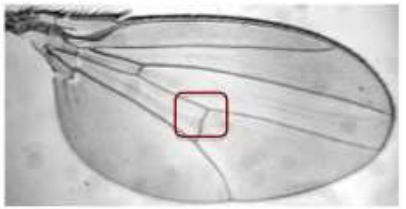
- state-dependent rates require special tool support
  -> observer variables

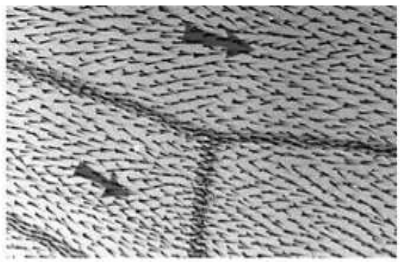
SUMMARY & OUTLOOK

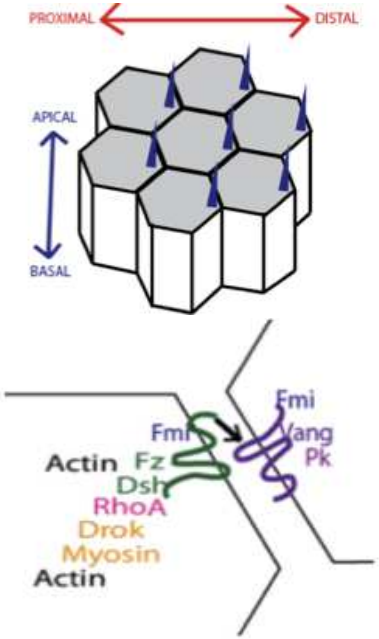
OUTLOOK

- **efficient simulation of very large Petri nets**
  - -> *stochastic*
  - -> *continuous*
  - -> *hybrid*
- **(hierarchical) space**
- **hierarchical organisation of components**
- **observables**
- **dynamic grid size**
- **shape and volume of components**
- **biosystem development**

Multiscale Challenges

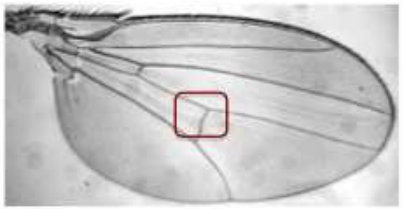


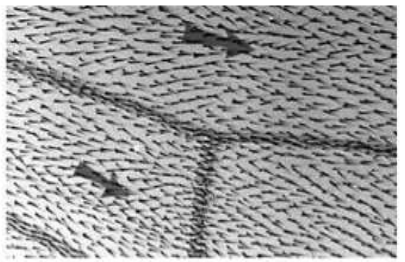


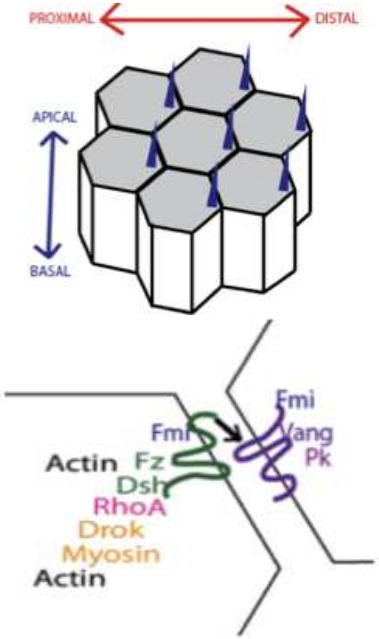


MULTISCALE CHALLENGES

- repetition of components
- variation of components
- organisation of components
- communication between components
- mobility / motility
- replication / deletion of components
- hierarchical organisation of components
- dynamic grid size
- irregular / semi-regular organisation of components

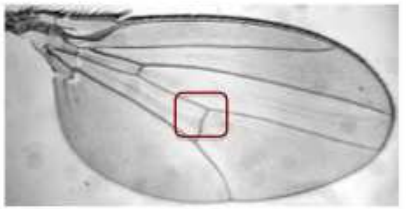


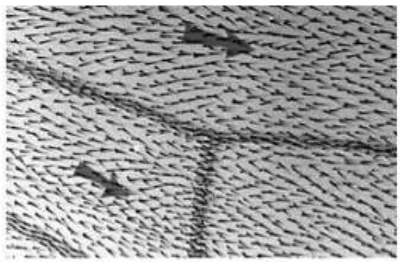


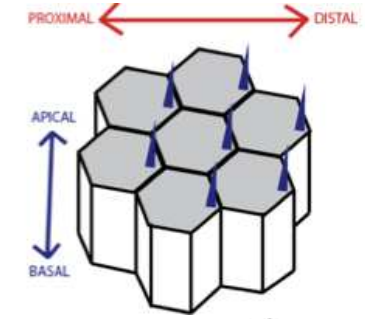


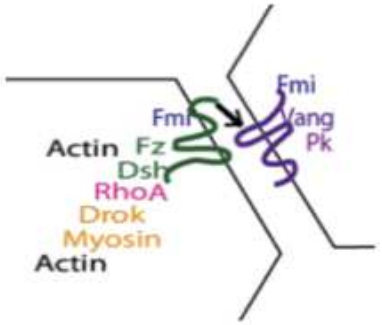
NOT COVERED TODAY

- **model checking**
  - -> *QPN - CTL*
  - -> *SPN - CSL, PLTLc*
  - -> *CPN - PLTLc*
- **advanced analysis techniques**
  - -> *image analysis*
  - -> *cluster analysis*
- **advanced modelling techniques**
  - -> *(semi-) automatic module generation*
  - -> *module mutation*
- **future work**
  - -> *protein (3D) shape and volume*
  - -> *biosystem development*
  - -> *hierarchical organisation of components*

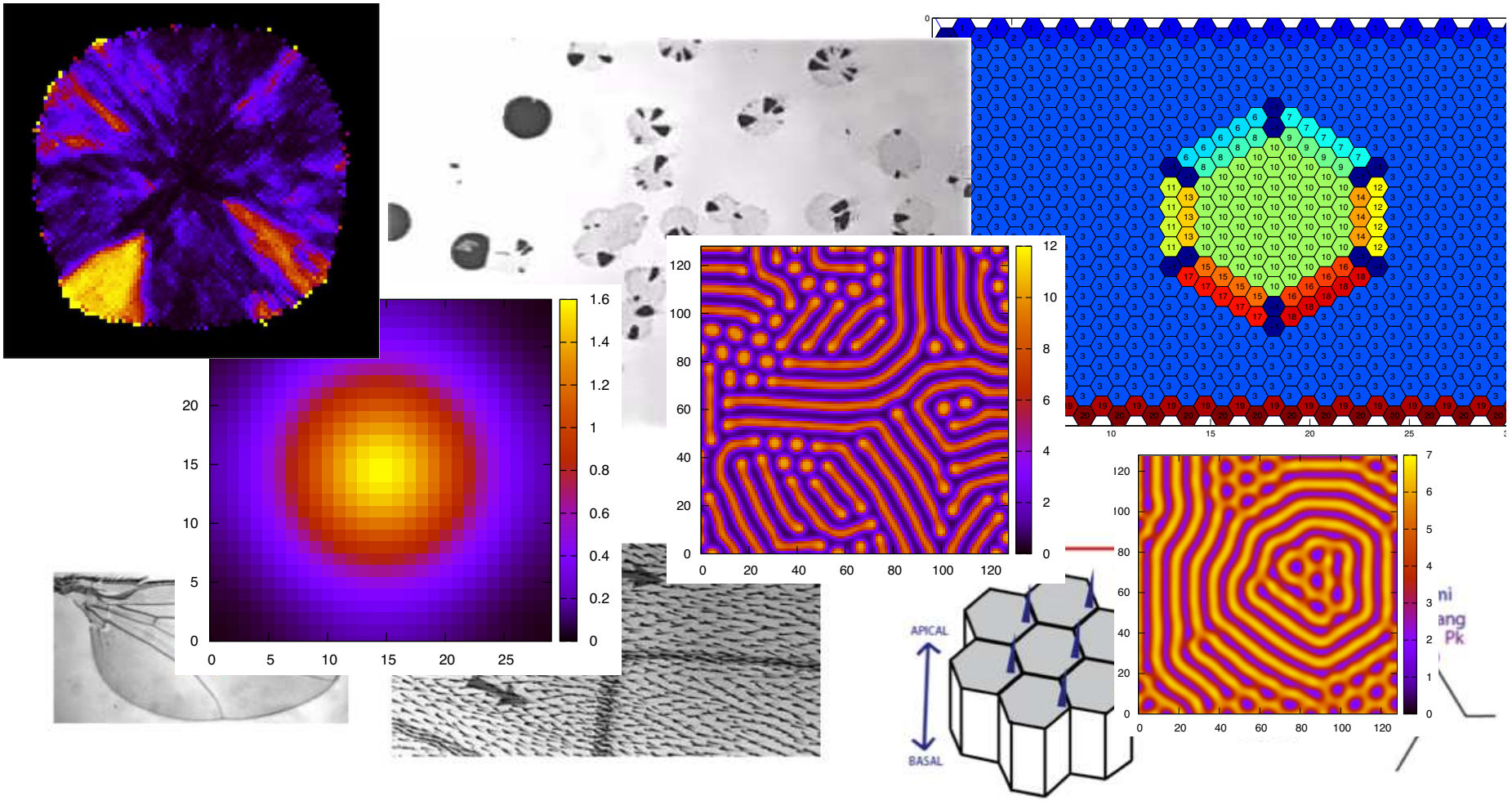








THANKS !



HTTP://MULTISCALEPN.BRUNEL.AC.UK