

Advanced Model Identification using Global Optimization

Toolbox brief description.

Eva Balsa-Canto



Process Engineering Group, IIM-CSIC

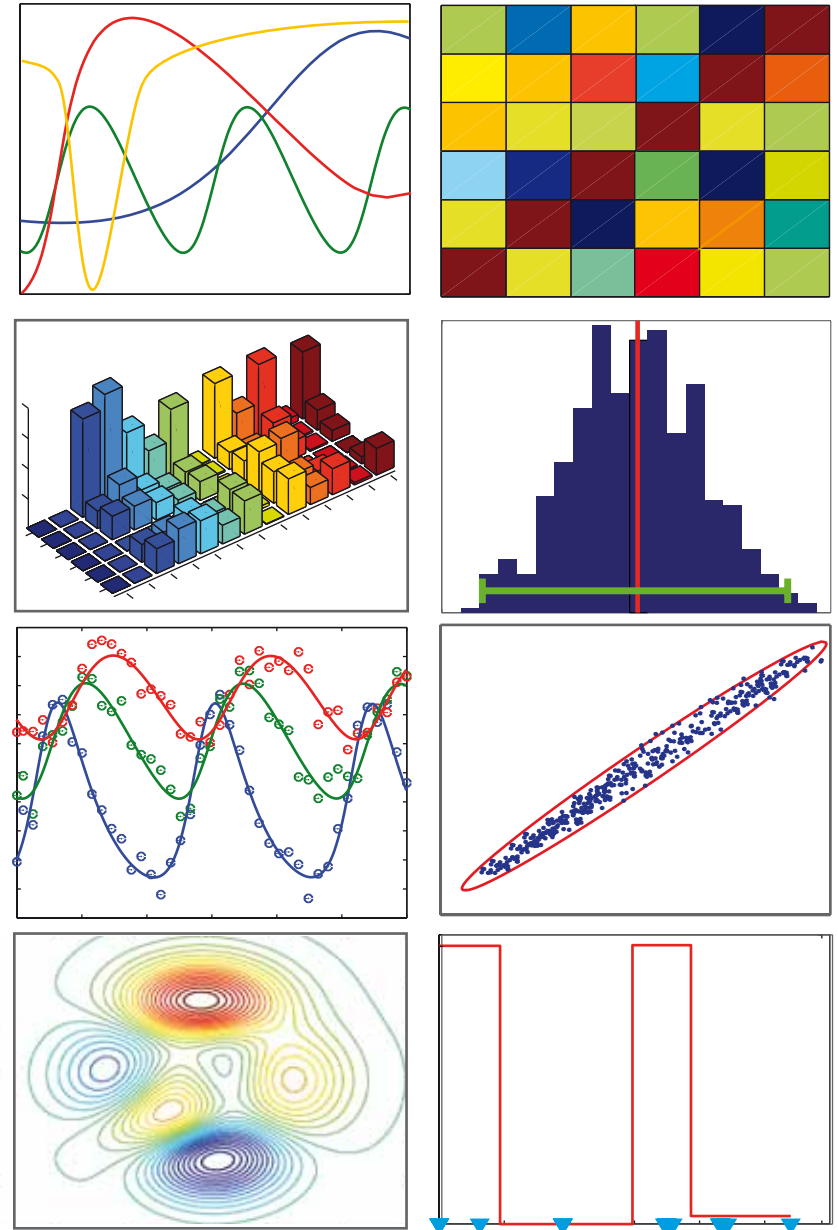
Vigo-Spain

*e-mail: ebalsa@iim.csic.es

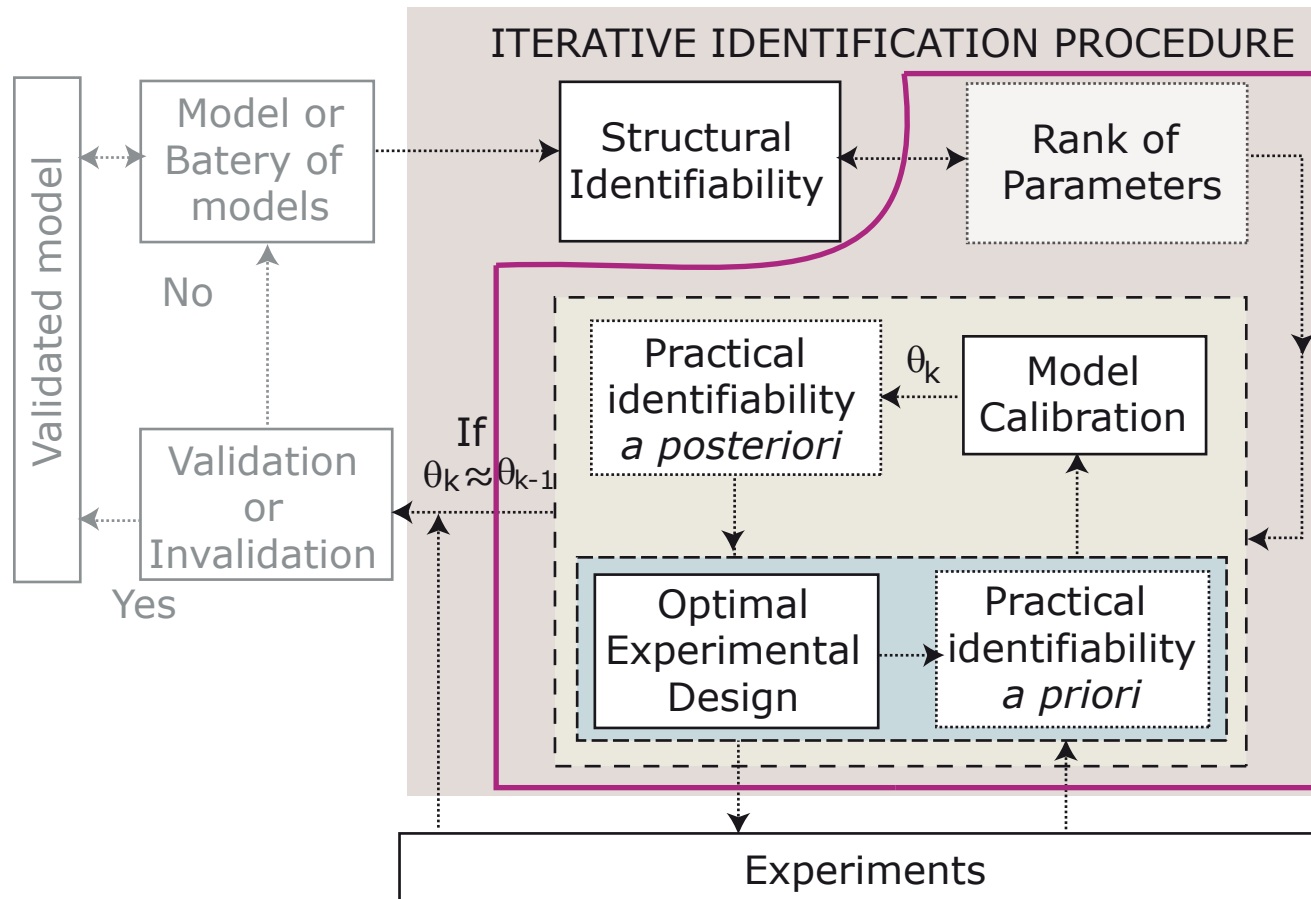


Outline

- > Scope of AMIGO
- > General structure & summary of features
- > Available tasks
 - > Simulation
 - > Experimental data
 - > Local and Global rank
 - > Parameter estimation
 - > Identifiability analysis
 - > Optimal experimental design



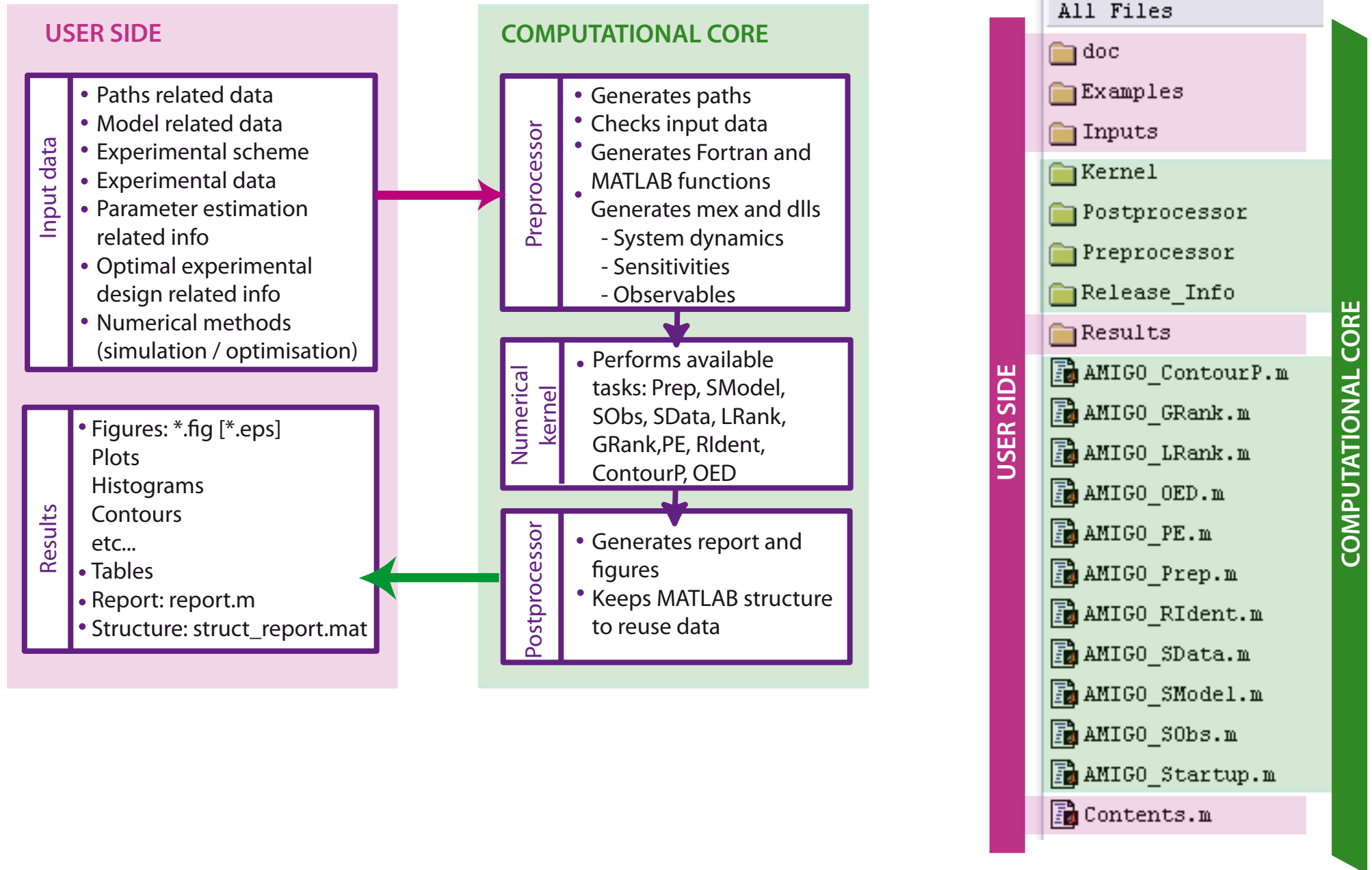
AMIGO is a MATLAB toolbox devoted to the Advanced Model Identification using Global Optimization methods. It is intended to cover most of the aspects in the following model identification loop:



- Balsa-Canto E, Alonso AA, Banga JR. An optimal identification procedure for model identification in systems biology: Applications in cell signalling. In: Algöwer & Reuss (Eds). Foundations of Systems Biology in Engineering. (2007)
- Balsa-Canto E, Alonso AA, Banga JR. Computational Procedures for Optimal Experimental Design in Biological Systems. IET Systems Biology 2(4):163-172. (2008)
- Banga JR, Balsa-Canto E. Parameter estimation and optimal experimental design. Essays in Biochemistry 45:195-210. (2008)
- Balsa-Canto E, Alonso AA, Banga JR. An iterative identification procedure for dynamic modeling of biochemical networks. BMC Systems Biology, 4:11 (2010).

General structure

The code is organised attending to the following scheme:



Summary of features (I)

Models

> Deterministic Dynamic models	Any non-linear general form
> Format	May be provided in MATLAB, FORTRAN, SBML, strings, black-box model
> Observation functions	Any linear / non-linear function of the states
> Notation	Customized names for states, parameters, stimuli & observables are allowed ('u' and 'v' are reserved)

Experimental data

> Definition of experimental scheme	Number of experiments, observables, initial conditions, sampling times, type of noise, stimuli. Flexibility over experiments.
> Pseudo-experimental data	"Numerical" data under experimental scheme conditions.
> Experimental data	Experimental time-series data plus error bars (if available).
> Stimuli	Theoretical or measured stimuli (if available).

Summary of features (I)

Models

> <i>Deterministic Dynamic models</i>	Any non-linear general form
> <i>Format</i>	May be provided in MATLAB, FORTRAN, SBML, strings, black-box model
> <i>Observation functions</i>	Any linear / non-linear function of the states
> <i>Notation</i>	Customized names for states, parameters, stimuli & observables are allowed ('u' and 'v' are reserved)

Experimental data

> <i>Definition of experimental scheme</i>	Number of experiments, observables, initial conditions, sampling times, type of noise, stimuli. Flexibility over experiments.
> <i>Pseudo-experimental data</i>	"Numerical" data under experimental scheme conditions.
> <i>Experimental data</i>	Experimental time-series data plus error bars (if available).
> <i>Stimuli</i>	Theoretical or measured stimuli (if available).

Summary of features (II)

IVP Solvers

> Non-Stiff and mildly stiff	RKF45	FORTTRAN	Runge-kutta-fehlberg (4,5) method: E. Fehlberg , Low-order classical Runge-Kutta formulas with stepsize control , NASA tr r-315
	ode113	MATLAB	Adams-Bashforth-Moulton (1,12): The MATLAB ODE Suite, L. F. Shampine & M. W. Reichelt, SIAM Journal on Scientific Computing, 18-1, (1997)
> Stiff	Radau5	FORTTRAN	Implicit Runge-Kutta Method: E. Hairer & G. Wanner, Solving ordinary differential equations II. Stiff and Differential-algebraic problems. Springer Series in Computational Mathematics 14, Springer-Verlag, 1996.
	LSODA	FORTTRAN	ADAMS with authomatic switch to BDF: A. C. Hindmarsh, ODEPACK, A systematized collection of ODE solvers, Scientific Computing, R. S. Stepleman et al. (eds.), Amsterdam, pp. 55-64 (1983) L.R. petzold, Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations, SIAM J. Sci. Stat. Comput. 4: 136-148.(1983)
	ode15s	MATLAB	Klopfenstein-Shampine BDF: The MATLAB ODE Suite, L. F. Shampine & M. W. Reichelt, SIAM Journal on Scientific Computing, 18-1, (1997)
> Sparse, Stiff	LSODES	FORTTRAN	BDF: A. C. Hindmarsh, ODEPACK, A systematized collection of ODE solvers, Scientific Computing, R. S. Stepleman et al. (eds.), Amsterdam, pp. 55-64 (1983) S.C. Eisenstat et al. Yale Sparse Matrix package. I & II. Int. J. Num. Meth. Eng., 18 (1982)
> To compute sensitivities	ODESSA	FORTTRAN	BDF: Leis JR, Kramer MA: Sensitivity Analysis of Systems of Differential and Algebraic Equations. Comp & Chem Eng 1985, 9(3):93-96.
	SENSMAT	MATLAB	Modification of ODE15s by V.M. García Mollá & R. Gómez Padilla to compute parametric sensitivities (2002).
	Finite Differences		http://www.mathworks.com/matlabcentral/fileexchange/1480-sensitivity-analysis-for-odes-and-daes

Summary of features (III)

NLP Solvers

> Local deterministic	Direct methods: nomad, dhc Indirect methods: fmincon, n2fb, dn2fb, ipopt, solnp, fsqp, misqp
> Multistart	Multistart of local solvers
> Global stochastic	Differential Evolution (DE), Stochastic Ranking Evolutionary Search (SRES)
> (Sequential) Hybrid methods	All possible combinations of Global stochastic methods with the above mentioned local solvers
> Metaheuristics	ssm and fssm

fmincon: SQP (Sequential Quadratic Programming), implemented as part of the Matlab optimization Toolbox

solnp: Y. Ye. Interior algorithms for linear, quadratic and linearly constrained non-linear programming. PhD, Stanford University, 1987..

fsqp: Panier and A. L. Tits. On combining feasibility, descent and superlinear convergence in inequality constrained optimization. *Math. Prog.*, 59:261-276, 1993.

ipopt: A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math Prog.*, 106(1):25-57, 2006.

misqp: O. Exler and K. Schittkowski. A trust region SQP algorithm for mixed-integer nonlinear programming. *Opt. Lett.*, 1(3):269-280, 2007.

n2fb: J.E. Dennis, D. M. Gay, and R. E. Welsch. An adaptive non-linear least-squares algorithm. *ACM Trans Math Soft*, 7(3):348-368, 1981.

NOMAD: M. A. Abramson. Pattern Search Algorithms for Mixed Variable General Constrained Optimization Problems. PhD, Rice University, 2002.

dhc: Dynamic Hill Climbing algorithm. M. de la Maza and D. Yuret. Dynamic hill climbing. *AI Expert*, 9(3):26-31, 1994.

DE: Storn R, Price K. Differential Evolution – a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *J Global Optim*, 11:341-359, (1997)

SRES: Runarsson T, Yao X. Stochastic ranking for constrained evolutionary optimization. *IEEE Trans Evol Comp*, 564:284-294, (2000)

Hybrids: Rodriguez-Fernandez M, Mendes P, Banga JR. A hybrid approach for efficient and robust parameter estimation in biochemical pathways. *Biosyst*, 83:248-265, (2006)

Metaheuristics: Egea JA, Rodriguez-Fernandez M, Banga JR, Martí R. Scatter Search for Chemical and Bio-Process Optimization. *J Glob Opt*, 37(3):481-503, (2007)

Available tasks

> <i>AMIGO initialization</i>	<i>AMIGO_Startup</i>
> <i>Model preprocessing</i>	<i>AMIGO_Prep</i>
> <i>Model simulation</i>	<i>AMIGO_SModel & AMIGO_SObs</i>
> <i>Experimental data</i>	<i>AMIGO_SData</i>
> <i>Rank of unknowns</i>	<i>AMIGO_LRank & AMIGO_GRank</i>
> <i>Unknowns estimation</i>	<i>AMIGO_PE</i>
> <i>Identifiability analysis</i>	<i>AMIGO_RIdent & AMIGO_ContourP</i>
> <i>Optimal experimental design</i>	<i>AMIGO_OED</i>

Simulation

- > **AMIGO_SModel** Simulates model under given experimental scheme and value of the vector of unknowns.
- > **AMIGO_SObs** Simulates observables under given experimental scheme and value of the vector of unknowns.

Inputs	Models	> Deterministic dynamic model Any non-linear general form > Format May be provided in MATLAB, FORTRAN, SBML, strings, black-box > Observation functions Any linear / non-linear function of the states > Notation Customized names for variables, states & observables are allowed
	Experimental Scheme	> Experiments info Flexible number of experiments Initial conditions [per experiment] Experiment duration [per experiment] Unknown values [per experiment] Observables [per experiment] <i>Only for AMIGO_SObs</i> [Stimuli conditions per experiment: sustained, step-wise, pulse-wise, linear interpolation]
	IVP Solvers	> Explicit Runge-Kutta type methods: RKF45; Adams method: ode113 Matlab > Implicit Methods for stiff systems: Radau5, ode15s Matlab, LSODA, LSODES > Programming Mex-files to FORTRAN compiled code are used to accelerate simulation
	Outputs	> Report .m file that keeps input data > Structure .mat that keeps inputs. and results. pathd, results.plotd and results.sim structures > Figures .fig [and .eps] files with the evolution of states or observables [and stimuli] vs time.

How to input a model in AMIGO

Illustrative example: The circadian clock in *Arabidopsis thaliana*

J.C.W. Locke, A.J. Millar, M.S. Turner. Modelling genetic networks with noisy and varied experimental data: the circadian clock in *Arabidopsis thaliana*. Journal of Theoretical Biology 234 (2005) 383-393

inputs.model.input_model_type='charmodelF' or 'charmodelM'

automatically generates a FORTRAN or a MATLAB model

```
% MODEL RELATED DATA
inputs.model.input_model_type='charmodelF';
inputs.model.n_st=7;
inputs.model.n_par=27;
inputs.model.n_stimulus=1;
inputs.model.st_names=char('CL_m','CL_c','CL_n','CT_m','CT_c','CT_n','CP_n');
inputs.model.par_names=char('n1','n2','g1','g2','m1','m2','m3','m4','m5',...
                             'm6','m7','k1','k2','k3','k4','k5','k6','k7',...
                             'p1','p2','p3','r1','r2','r3','r4','q1','q2');
inputs.model.stimulus_names=char('light');
inputs.model.eqns=...
char('dCL_m=q1*CP_n*light+n1*CT_n/(g1+CT_n)-m1*CL_m/(k1+CL_m)',...
     'dCL_c=p1*CL_m-r1*CL_c+r2*CL_n-m2*CL_c/(k2+CL_c)',...
     'dCL_n=r1*CL_c-r2*CL_n-m3*CL_n/(k3+CL_n)',...
     'dCT_m=n2*g2**2/(g2**2+CL_n**2)-m4*CT_m/(k4+CT_m)',...
     'dCT_c=p2*CT_m-r3*CT_c+r4*CT_n-m5*CT_c/(k5+CT_c)',...
     'dCT_n=r3*CT_c-r4*CT_n-m6*CT_n/(k6+CT_n)',...
     'dCP_n=(1-light)*p3-m7*CP_n/(k7+CP_n)-q2*light*CP_n');
```

inputs.model.input_model_type='fortranmodel'

to input fortran files to compute system dynamics and sensitivities

```
% MODEL RELATED DATA
inputs.model.input_model_type='fortranmodel';
inputs.model.fortranmodel_file='fcn_circadian'; % File including the system dynamics
inputs.model.fortransens_file='sens_circadian'; % File to compute sensitivities
inputs.model.n_st=7; inputs.model.n_par=27; inputs.model.n_stimulus=1;
inputs.model.st_names=char('CL_m','CL_c','CL_n','CT_m','CT_c','CT_n','CP_n');
inputs.model.par_names=char('n1','n2','g1','g2','m1','m2','m3','m4','m5',...
                             'm6','m7','k1','k2','k3','k4','k5','k6','k7',...
                             'p1','p2','p3','r1','r2','r3','r4','q1','q2');
inputs.model.stimulus_names=char('light');
```

inputs.model.input_model_type='matlabmodel'

to input a matlab file with odes

```
% MODEL RELATED DATA
inputs.model.input_model_type='matlabmodel';
inputs.model.matlabmodel_file='odes_circadian'; % File including the system dynamics
inputs.model.n_st=7; inputs.model.n_par=27; inputs.model.n_stimulus=1;
inputs.model.st_names=char('CL_m','CL_c','CL_n','CT_m','CT_c','CT_n','CP_n');
inputs.model.par_names=char('n1','n2','g1','g2','m1','m2','m3','m4','m5',...
                             'm6','m7','k1','k2','k3','k4','k5','k6','k7',...
                             'p1','p2','p3','r1','r2','r3','r4','q1','q2');
inputs.model.stimulus_names=char('light');
```

inputs.model.input_model_type='sbmlmodel'

to input a sbml file with odes

```
% MODEL RELATED DATA
inputs.model.input_model_type='sbmlmodel';
inputs.model.matlabmodel_file='BIOMD0000000005';
inputs.model.n_st=7; inputs.model.n_par=27; inputs.model.n_stimulus=1;
inputs.model.st_names=char('CL_m','CL_c','CL_n','CT_m','CT_c','CT_n','CP_n');
inputs.model.par_names=char('n1','n2','g1','g2','m1','m2','m3','m4','m5',...
                             'm6','m7','k1','k2','k3','k4','k5','k6','k7',...
                             'p1','p2','p3','r1','r2','r3','r4','q1','q2');
inputs.model.stimulus_names=char('light');
```

inputs.model.input_model_type='blackboxmodel'

to input a black box model which performs a simulation for a given experiment

```
% MODEL RELATED DATA
inputs.model.input_model_type='blackboxmodel';
inputs.model.blackboxmodel_file='circadianbbmodel.m';
inputs.model.n_st=7; inputs.model.n_par=27; inputs.model.n_stimulus=1;
inputs.model.st_names=char('CL_m','CL_c','CL_n','CT_m','CT_c','CT_n','CP_n');
inputs.model.par_names=char('n1','n2','g1','g2','m1','m2','m3','m4','m5',...
                             'm6','m7','k1','k2','k3','k4','k5','k6','k7',...
                             'p1','p2','p3','r1','r2','r3','r4','q1','q2');
inputs.model.stimulus_names=char('light');
```

Simulation: AMIGO_SModel

Inputs

```
% PATHS RELATED DATA
results.pathd.results_folder='circadian-tutorial';
results.pathd.short_name='circadian';

% MODEL RELATED DATA
inputs.model.input_model_type='chamodelF';
inputs.model.n_st=7;
inputs.model.n_par=27;
inputs.model.n_stimulus=1;
inputs.model.st_names=char('CL_m','CL_c','CL_n','CT_m','CT_c','CT_n','CP_n');
inputs.model.par_names=char('n1','n2','g1','g2','m1','m2','m3','m4','m5',...
    'm6','m7','k1','k2','k3','k4','k5','k6','k7',...
    'p1','p2','p3','r1','r2','r3','r4','q1','q2');
inputs.model.stimulus_names=char('light');
inputs.model.eqns=...
char('dCL_m=q1*CP_n*light+n1*CT_n/(g1+CT_n)-m1*CL_m/(k1+CL_m)',...
    'dCL_c=p1*CL_m-r1*CL_c+r2*CL_n-m2*CL_c/(k2+CL_c)',...
    'dCL_n=r1*CL_c-r2*CL_n-m3*CL_n/(k3+CL_n)',...
    'dCT_m=n2*g2**2/(g2**2+CL_n**2)-m4*CT_m/(k4+CT_m)',...
    'dCT_c=p2*CT_m-r3*CT_c+r4*CT_n-m5*CT_c/(k5+CT_c)',...
    'dCT_n=r3*CT_c-r4*CT_n-m6*CT_n/(k6+CT_n)',...
    'dCP_n=(1-light)*p3-m7*CP_n/(k7+CP_n)-q2*light*CP_n');
inputs.model.par=[7.5038 0.6801 1.4992 3.0412 10.0982 1.9685 3.7511 2.3422 ...
    7.2482 1.8981 1.2 3.8045 5.3087 4.1946 2.5356 1.4420 4.8600 ...
    1.2 2.1994 9.4440 0.5 0.2817 0.7676 0.4364 7.3021 4.5703 1.0];

% EXPERIMENTAL SCHEME RELATED DATA
inputs.exps.n_exp=2;
for iexp=1:inputs.exps.n_exp
    inputs.exps.exp_y0{iexp}=zeros(1,inputs.model.n_st);
    inputs.exps.t_f{iexp}=120;
end
inputs.exps.u_interp{1}='sustained';
inputs.exps.t_con{1}=[0 120];
inputs.exps.u{1}=[1];
inputs.exps.u_interp{2}='pulse-down';
inputs.exps.n_pulses{2}=5;
inputs.exps.t_con{2}=[0 :12: 120];
```

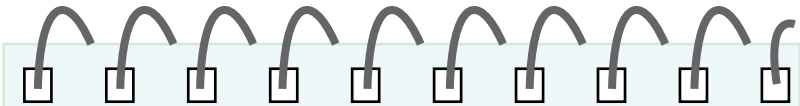
Minimum input data for simulation.

Illustrative example: The circadian clock in *Arabidopsis thaliana*

J.C.W. Locke, A.J. Millar, M.S. Turner. Modelling genetic networks with noisy and varied experimental data: the circadian clock in *Arabidopsis thaliana*. Journal of Theoretical Biology 234 (2005) 383-393

Simulation of two experiments:

- > 1st under sustained stimulation
- > 2nd under pulse-wise stimulation.



How to simulate a model with AMIGO?

```
% From AMIGO path....
% Initialize AMIGO to begin its use

>> AMIGO_Startup

% Preprocess the problem model. This is
% required only once, unless model changes
% are incorporated.

>> AMIGO_Prep('circadian_smodel')

% Simulate

>> AMIGO_SModel('circadian_smodel')
```

Simulation: AMIGO_SModel

Outputs

....\AMIGO\Results

circadian-tutorial

SModel_circadian_run1

AMIGO_gen_obs_circadian.m
fcn.f
sens.f

Files generated during preprocessing

circadian_smodel_input_run1.m
report_circadian_run1.m
states_plot_exp1.fig
states_plot_exp2.fig
strreport_circadian_run1.mat

```
>> load strreport_circadian_run1.mat  
>> who  
Your variables are:  
inputs results
```

```
>> inputs  
  
pathd: [1x1 struct]  
model: [1x1 struct]  
exps: [1x1 struct]  
ivpsol: [1x1 struct]  
input_file: 'circadian_smodel'
```

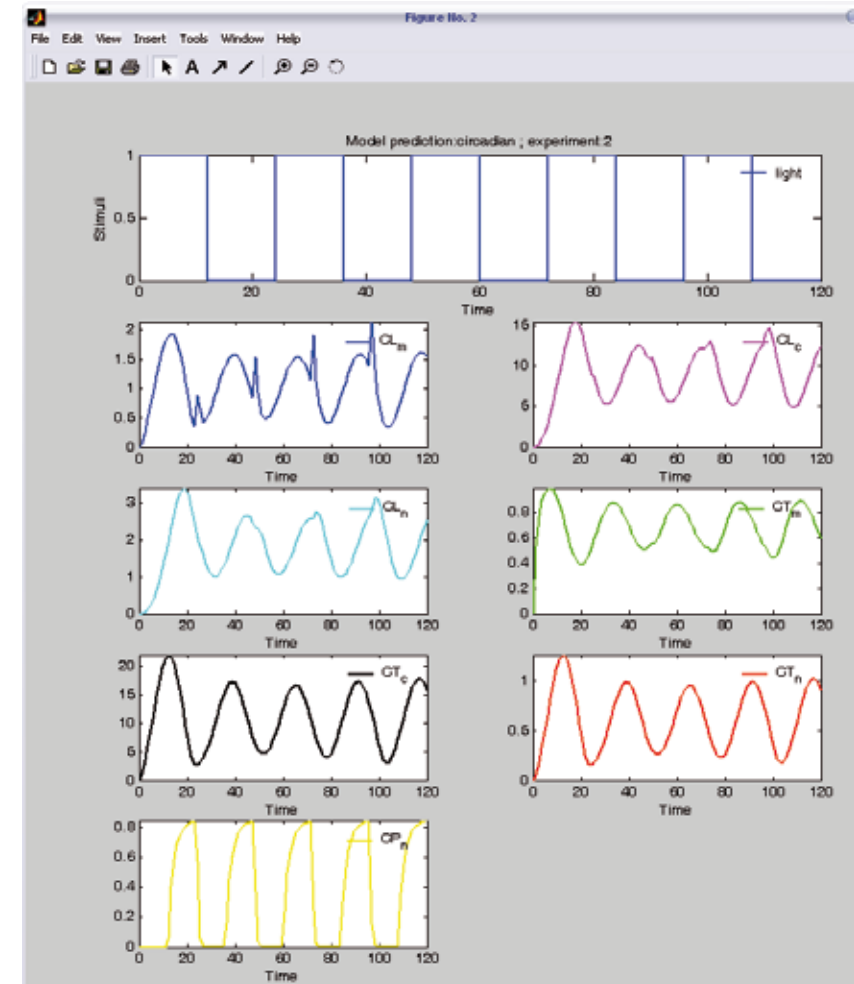
```
>> results
```

```
pathd: [1x1 struct]  
plotd: [1x1 struct]  
sim: [1x1 struct]
```

```
>> results.sim
```

```
tsim: {[1x100 double] [1x100 double]}  
states: {[100x7 double] [100x7 double]}
```

Illustrative example: The circadian clock in Arabidopsis thaliana



states_plot_exp2.fig

Simulation: AMIGO_SObs

Inputs

```
% PATHS RELATED DATA
results.pathd.results_folder='circadian-tutorial';
results.pathd.short_name='circadian';

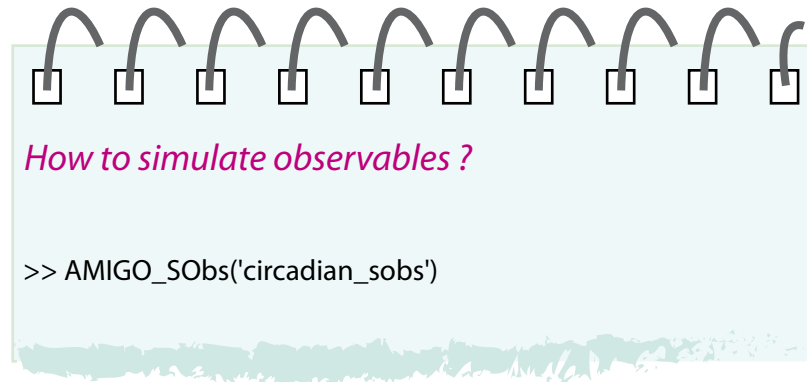
% MODEL RELATED DATA
inputs.model.input_model_type='charmodelF';
inputs.model.n_st=7; inputs.model.n_par=27; inputs.model.n_stimulus=1;
inputs.model.st_names=char('CL_m','CL_c','CL_n','CT_m','CT_c','CT_n','CP_n');
inputs.model.par_names=char('n1','n2','g1','g2','m1','m2','m3','m4','m5',...
                             'm6','m7','k1','k2','k3','k4','k5','k6','k7',...
                             'p1','p2','p3','r1','r2','r3','r4','q1','q2');

inputs.model.stimulus_names=char('light');
inputs.model.eqns=...
char('dCL_m=q1*CP_n*light+n1*CT_n/(g1+CT_n)-m1*CL_m/(k1+CL_m)',...
     'dCL_c=p1*CL_m-r1*CL_c+r2*CL_n-m2*CL_c/(k2+CL_c)',...
     'dCL_n=r1*CL_c-r2*CL_n-m3*CL_n/(k3+CL_n)',...
     'dCT_m=n2*g2**2/(g2+CL_n**2)-m4*CT_m/(k4+CT_m)',...
     'dCT_c=p2*CT_m-r3*CT_c+r4*CT_n-m5*CT_c/(k5+CT_c)',...
     'dCT_n=r3*CT_c-r4*CT_n-m6*CT_n/(k6+CT_n)',...
     'dCP_n=(1-light)*p3-m7*CP_n/(k7+CP_n)-q2*light*CP_n');
inputs.model.par=[7.5038 0.6801 1.4992 3.0412 10.0982 1.9685 3.7511 2.3422 ...
                  7.2482 1.8981 1.2 3.8045 5.3087 4.1946 2.5356 1.4420 4.8600 ...
                  1.2 2.1994 9.4440 0.5 0.2817 0.7676 0.4364 7.3021 4.5703 1.0];

% EXPERIMENTAL SCHEME RELATED DATA
inputs.exps.n_exp=2;
for iexp=1:inputs.exps.n_exp
inputs.exps.exp_y0{iexp}=zeros(1,inputs.model.n_st);
inputs.exps.t_f{iexp}=120;
inputs.exps.n_obs{iexp}=2;
inputs.exps.obs_names{iexp}=char('Lum','mRNAa');
inputs.exps.obs{iexp}=char('Lum=CL_m','mRNAa=CT_m');
end
inputs.exps.u_interp{1}='sustained';
inputs.exps.t_con{1}=[0 120];
inputs.exps.u{1}=[1];
inputs.exps.u_interp{2}='pulse-down';
inputs.exps.n_pulses{2}=5;
inputs.exps.t_con{2}=[0 :12: 120];
```

Illustrative example: The circadian clock in Arabidopsis thaliana

**Simulating 2 observables: Luminiscence and RNA amplitude
for two experiments: 1st under sustained stimulation
and 2nd under pulse-wise stimulation.**



Problems may be run from any path.
However results obtained will be ALWAYS
kept in the Results folder in AMIGO
(or a user defined folder within AMIGO path).

Simulation: AMIGO_SObS

Outputs

....\AMIGO\Results

circadian-tutorial

SModel_circadian_run1

SObs_circadian_run1

AMIGO_gen_obs_circadian.m

fcn.f

sens.f

circadian_sobs_input_run1.m

obs_plot_exp1.fig

obs_plot_exp2.fig

report_circadian_run1.m

strreport_circadian_run1.mat

```
>> load strreport_circadian_run1.mat
```

```
>> inputs
```

```
inputs =
```

```
    pathd: [1x1 struct]
```

```
    model: [1x1 struct]
```

```
    exps: [1x1 struct]
```

```
    ivpsol: [1x1 struct]
```

```
    input_file: 'circadian_sobs'
```

```
>> results
```

```
results =
```

```
    pathd: [1x1 struct]
```

```
    plotd: [1x1 struct]
```

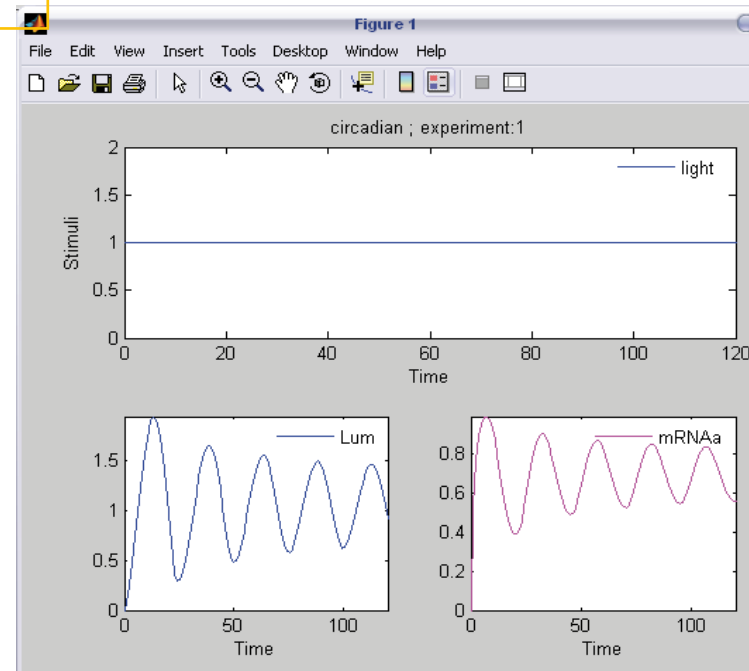
```
    sim: [1x1 struct]
```

```
>> results.sim
```

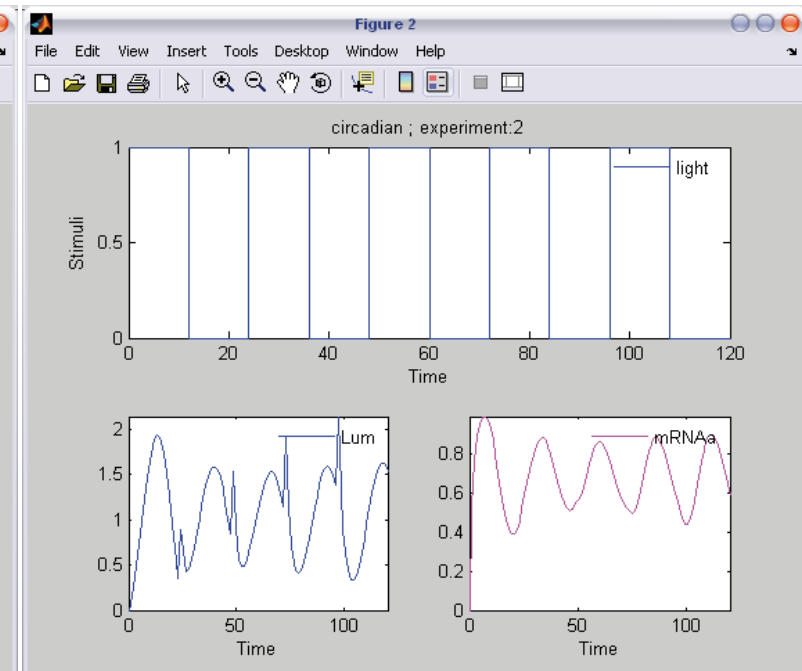
```
    tsim: {[1x100 double] [1x100 double]}
```

```
    states: {[100x7 double] [100x7 double]}
```

```
    obs: {[100x2 double] [100x2 double]}
```



obs_plot_exp1_1.fig



obs_plot_exp2_1.fig

Illustrative example: The circadian clock in Arabidopsis thaliana

Experimental data

- > **AMIGO_SData** Simulates observables and plots results vs experimental data or generates pseudo-experimental data for numerical tests

Inputs	Models	> Deterministic ODEs Any non-linear general form > Format May be provided in MATLAB, FORTRAN, SBML, strings, black-box > Observation functions Any linear / non-linear function of the states > Notation Customized names for variables, states & observables are allowed
	Experimental Data	> Experimental Scheme Flexible number of experiments; Initial conditions [per experiment] Experiment duration [per experiment]; Unknown values [per experiment] Observables [per experiment] ; [Stimuli conditions per experiment: sustained, step-wise, pulse-wise, linear interpolation]; Sampling times [per experiment] > Real data Matrices of experimental data and error bars if available > Experimental error info Normal error assumed: homoscedastic (constant or variable variance) heteroscedastic (power in the mean)
	IVP Solvers	> Explicit Runge-Kutta type methods: RKF45; Adams method: ode113 Matlab > Implicit Methods for stiff systems: Radau5, ode15s Matlab, LSODA, LSODES > Programming Mex-files to FORTRAN compiled code are used to accelerate simulation
Outputs		> Report .m file that keeps input data > Structure .mat that keeps inputs. and results.pathd, results.plotd, results.sim structures > Figures .fig [and .eps] files with the evolution of observables and experimental data [and stimuli] vs time.

Experimental data: pseudo-data

Inputs

```
% PATHS RELATED DATA
results.pathd.results_folder='circadian-tutorial';
results.pathd.short_name='circadian';

% MODEL RELATED DATA
inputs.model.input_model_type='chamodelF';
inputs.model.n_st=7; inputs.model.n_par=27; inputs.model.n_stimulus=1;
inputs.model.st_names=char('CL_m','CL_c','CL_n','CT_m','CT_c','CT_n','CP_n');
inputs.model.par_names=char('n1','n2','g1','g2','m1','m2','m3','m4','m5',...
                             'm6','m7','k1','k2','k3','k4','k5','k6','k7',...
                             'p1','p2','p3','r1','r2','r3','r4','q1','q2');

inputs.model.stimulus_names=char('light');
inputs.model.eqns=...
char('dCL_m=q1*CP_n*light+n1*CT_n/(g1+CT_n)-m1*CL_m/(k1+CL_m)',...
     'dCL_c=p1*CL_m-r1*CL_c+r2*CL_n-m2*CL_c/(k2+CL_c)',...
     'dCL_n=r1*CL_c-r2*CL_n-m3*CL_n/(k3+CL_n)',...
     'dCT_m=n2*g2**2/(g2+CL_n**2)-m4*CT_m/(k4+CT_m)',...
     'dCT_c=p2*CT_m-r3*CT_c+r4*CT_n-m5*CT_c/(k5+CT_c)',...
     'dCT_n=r3*CT_c-r4*CT_n-m6*CT_n/(k6+CT_n)',...
     'dCP_n=(1-light)*p3-m7*CP_n/(k7+CP_n)-q2*light*CP_n');
inputs.model.par=[7.5038 0.6801 1.4992 3.0412 10.0982 1.9685 3.7511 2.3422 ...
                  7.2482 1.8981 1.2 3.8045 5.3087 4.1946 2.5356 1.4420 4.8600 ...
                  1.2 2.1994 9.4440 0.5 0.2817 0.7676 0.4364 7.3021 4.5703 1.0];

% EXPERIMENTAL SCHEME RELATED DATA
inputs.exps.n_exp=2;
for iexp=1:inputs.exps.n_exp
inputs.exps.exp_y0{iexp}=zeros(1,inputs.model.n_st);
inputs.exps.t_f{iexp}=120;
inputs.exps.n_obs{iexp}=2;
inputs.exps.obs_names{iexp}=char('Lum','mRNAa');
inputs.exps.obs{iexp}=char('Lum=CL_m','mRNAa=CT_m');
end
inputs.exps.u_interp{1}='sustained';
inputs.exps.t_con{1}=[0 120];
inputs.exps.u{1}=[1];
inputs.exps.u_interp{2}='pulse-down';
inputs.exps.n_pulses{2}=5;
inputs.exps.t_con{2}=[0 :12: 120];

% EXPERIMENTAL DATA RELATED INFO
inputs.exps.data_type='pseudo_pos';
inputs.exps.noise_type='homo_var';
inputs.exps.n_s{1}=15;
inputs.exps.n_s{2}=25;
for iexp=1:inputs.exps.n_exp
inputs.exps.std_dev{iexp}=0.05;
end
```

Illustrative example: The circadian clock in Arabidopsis thaliana

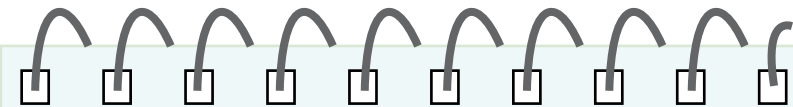
Generating pseudo-experimental data for numerical tests:

Two experiments:

1st under sustained stimulation, 15 equidistant sampling times

2nd under pulse-wise stimulation, 25 equidistant sampling times

Both experiments subject to gaussian experimental noise of 0 mean and varying variance (homoscedastic noise).



How to generate pseudo-experimental data ?

```
>> AMIGO_SData('circadian_pdata')
```

Experimental data: real data

Inputs

```
% PATHS RELATED DATA
.... as in circadian_sobs
% MODEL RELATED DATA
.... as in circadian_sobs
% EXPERIMENTAL SCHEME RELATED DATA
.... as in circadian_sobs

% EXPERIMENTAL DATA RELATED INFO
inputs.exps.data_type='real';
inputs.exps.noise_type='homo_var';
inputs.exps.n_s{1}=15;
inputs.exps.n_s{2}=25;
inputs.exps.exp_data{1}=[ 0.1405  0.1219
                        5.4326  1.4396
                        .....
                        3.4017  0.0382
                        0.2889  0.0819];
inputs.exps.error_data{1}=[ 0.1405  0.1219
                           0.2705  0.0285
                           .....
                           0.2041  0.0600
                           0.2711  0.0312];
inputs.exps.exp_data{2}=[ 0.4855  0.0741
                        3.8695  4.0924
                        6.7028  0.2161
                        6.4097  0.0957
                        .....
                        3.4529  0.5495
                        2.9209  0.0830
                        0.0660  0.1571];
inputs.exps.error_data{2}=[ 0.4855  0.0741
                           0.2213  0.1845
                           0.6119  0.4143
                           0.1577  0.0654
                           .....
                           0.3850  0.1784
                           0.3300  0.0109
                           0.0354  0.0979];
```



Matrices of: $n_s^{o,\varepsilon} \times n^{o,\varepsilon}$ elements can be incorporated for both the experimental data and the experimental error when available.

Matrices can be read from a .mat file or from .m files

Illustrative example: The circadian clock in Arabidopsis thaliana

Introducing experimental data plus corresponding experimental error:

Two experiments:

1st under sustained stimulation, 15 equidistant sampling times

2nd under pulse-wise stimulation, 25 equidistant sampling times

Both experiments subject to gaussian experimental noise of 0 mean and varying variance (homoscedastic noise).



How to check experimental data ?

```
>> AMIGO_SData('circadian_tutorial_rData')
```

Simulation: AMIGO_SData

Outputs

....\AMIGO\Results

circadian-tutorial

- SData_circadian_real
- SData_circadian_pseudo
- SModel_circadian_run1
- SObs_circadian_run1

- AMIGO_gen_obs_circadian.m
- fcn.f
- sens.f

- circadian_rdata_input_real.m
- data_plot_exp1.fig
- data_plot_exp2.fig
- report_circadian_real.m
- strreport_circadian_real.mat

```
>> load strreport_circadian_real.mat
```

Real data

```
>> results
```

```
pathd: [1x1 struct]
plotd: [1x1 struct]
sim: [1x1 struct]
```

```
>> results.sim
```

```
tsim: {[1x100 double] [1x100 double]}
states: {[100x7 double] [100x7 double]}
obs: {[100x2 double] [100x2 double]}
```

```
>> load strreport_circadian_pseudo.mat
```

Pseudo data

```
>> results
```

```
results =
```

```
pathd: [1x1 struct]
plotd: [1x1 struct]
sim: [1x1 struct]
fit: [1x1 struct]
```

```
>> results.sim
```

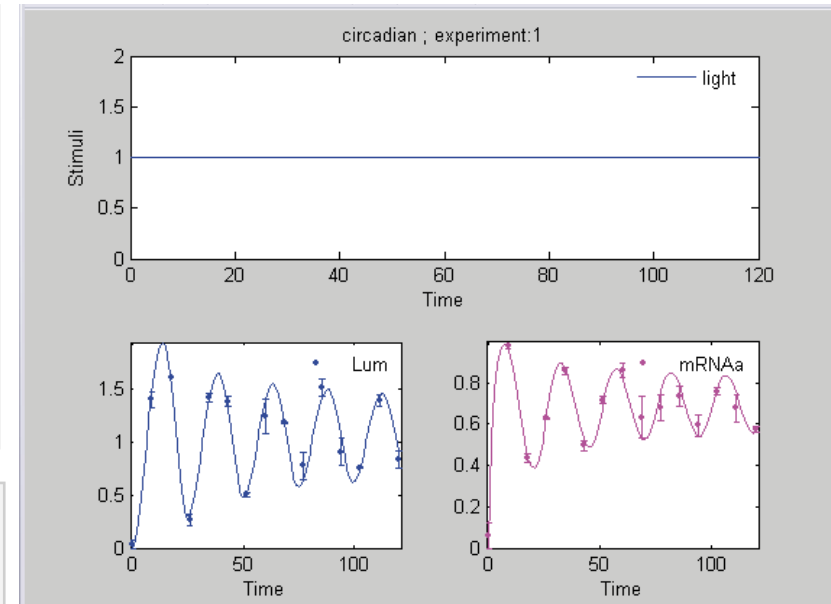
```
exp_data: {[15x2 double] [25x2 double]}
tsim: {[1x100 double] [1x100 double]}
states: {[100x7 double] [100x7 double]}
obs: {[100x2 double] [100x2 double]}
```

```
>> results.fit
```

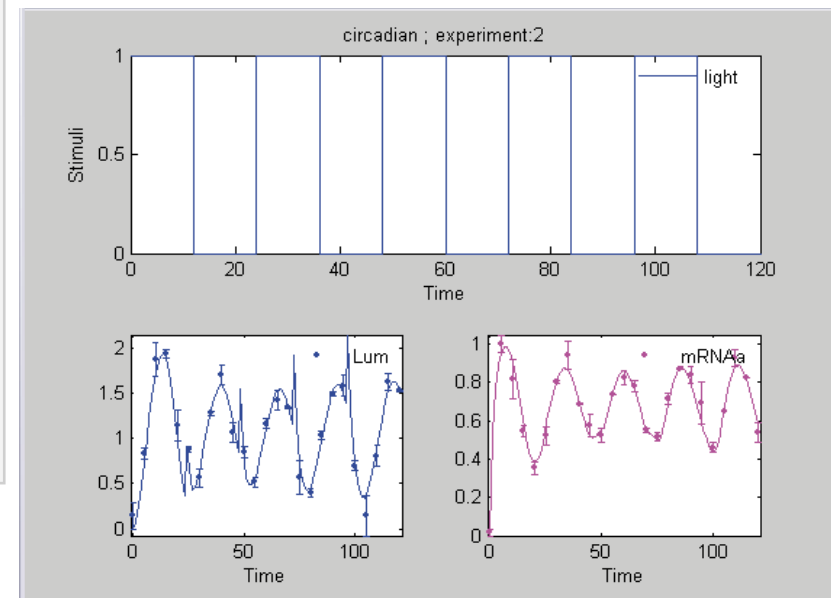
```
ans =
```

```
residuals: {[15x2 double] [25x2 double]}
norm_residuals: {[15x2 double] [25x2 double]}
```

Illustrative example: The circadian clock in Arabidopsis thaliana



data_plot_exp1.fig



data_plot_exp2.fig

Local & Global Rank

- > **AMIGO_LRank** Computes local parametric sensitivities and the rank of parameters for a given value of the vector of unknowns.
- > **AMIGO_GRank** Computes sensitivities and rank of parameters within a hyperrectangle (allowed values for the unknowns).

Inputs	Models	> Deterministic ODEs > Format > Observation functions > Notation	Any non-linear general form May be provided in MATLAB, FORTRAN, SBML, strings, black-box Any linear / non-linear function of the states Customized names for variables, states & observables are allowed
	Experimental Scheme	> Experiments info	Flexible number of experiments Initial conditions [per experiment, for rank per experiments] Experiment duration [per experiment] Unknown values [per experiment, for rank per experiments] Observables [per experiment] [Stimuli conditions per experiment: sustained, step-wise, pulse-wise, linear interpolation]
	IVP Solvers	> Explicit > Implicit > Programming	Runge-Kutta type methods: RKF45; Adams method: ode113 Matlab Methods for stiff systems: Radau5, ode15s Matlab, LSODA, LSODES To compute sensitivities: ODESA, SENS_SYS (for MATLAB), Finite Differences Mex-files to FORTRAN compiled code are used to accelerate simulation
	Outputs	> Report > Structure > Figures	.m file that keeps input data and rank results .mat that keeps inputs. and results. pathd, .plotd, .sim, .rank .fig [and .eps] files with the evolution of local sensitivities vs time (optional), local or global ranking of parameters, local or global sensitivities for each observable and experiment (bar and surface plots)

Local Rank

Inputs

```
% PATHS RELATED DATA
results.pathd.results_folder_name='circadian-tutorial';
results.pathd.short_name='circadian';

% MODEL RELATED DATA
inputs.model.input_model_type='charmodelF';
inputs.model.n_st=7; inputs.model.n_par=27; inputs.model.n_stimulus=1;
inputs.model.st_names=char('CL_m','CL_c','CL_n','CT_m','CT_c','CT_n','CP_n');
inputs.model.par_names=char('n1','n2','g1','g2','m1','m2','m3','m4','m5',...
                             'm6','m7','k1','k2','k3','k4','k5','k6','k7',...
                             'p1','p2','p3','r1','r2','r3','r4','q1','q2');

inputs.model.stimulus_names=char('light');
inputs.model.eqns=...
char('dCL_m=q1*CP_n*light+n1*CT_n/(g1+CT_n)-m1*CL_m/(k1+CL_m)',...
     'dCL_c=p1*CL_m-r1*CL_c+r2*CL_n-m2*CL_c/(k2+CL_c)',...
     'dCL_n=r1*CL_c-r2*CL_n-m3*CL_n/(k3+CL_n)',...
     'dCT_m=n2*g2**2/(g2+CL_n**2)-m4*CT_m/(k4+CT_m)',...
     'dCT_c=p2*CT_m-r3*CT_c+r4*CT_n-m5*CT_c/(k5+CT_c)',...
     'dCT_n=r3*CT_c-r4*CT_n-m6*CT_n/(k6+CT_n)',...
     'dCP_n=(1-light)*p3-m7*CP_n/(k7+CP_n)-q2*light*CP_n');
inputs.model.par=[7.5038 0.6801 1.4992 3.0412 10.0982 1.9685 3.7511 2.3422 ...
                  7.2482 1.8981 1.2 3.8045 5.3087 4.1946 2.5356 1.4420 4.8600 ...
                  1.2 2.1994 9.4440 0.5 0.2817 0.7676 0.4364 7.3021 4.5703 1.0];

% EXPERIMENTAL SCHEME RELATED DATA
inputs.exps.n_exp=2;
for iexp=1:inputs.exps.n_exp
inputs.exps.exp_y0{iexp}=zeros(1,inputs.model.n_st);inputs.exps.t_ff{iexp}=120;
end
inputs.exps.u_interp{1}='sustained'; inputs.exps.t_con{1}=[0 120];
inputs.exps.u{1}=[1];
inputs.exps.u_interp{2}='pulse-down'; inputs.exps.n_pulses{2}=5;
inputs.exps.t_con{2}=[0 :12: 120];

% EXPERIMENTAL DATA RELATED INFO
inputs.exps.n_s{1}=15; inputs.exps.n_s{2}=25;

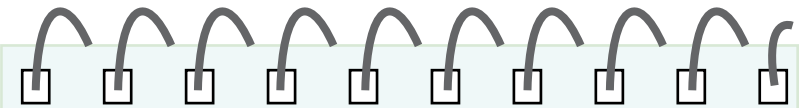
% UNKNOWNNS RELATED DATA
% To rank all parameters:
inputs.PEsol.id_global_theta='all';
% To rank selected parameters:
inputs.PEsol.id_global_theta= char('g1','m4','m5','k2','k3','p1');
```

Illustrative example: The circadian clock in Arabidopsis thaliana

Local sensitivities and rank of parameters.

Two experiments:

1st under sustained stimulation, 15 equidistant sampling times
2nd under pulse-wise stimulation, 25 equidistant sampling times



How to rank the parameters ?

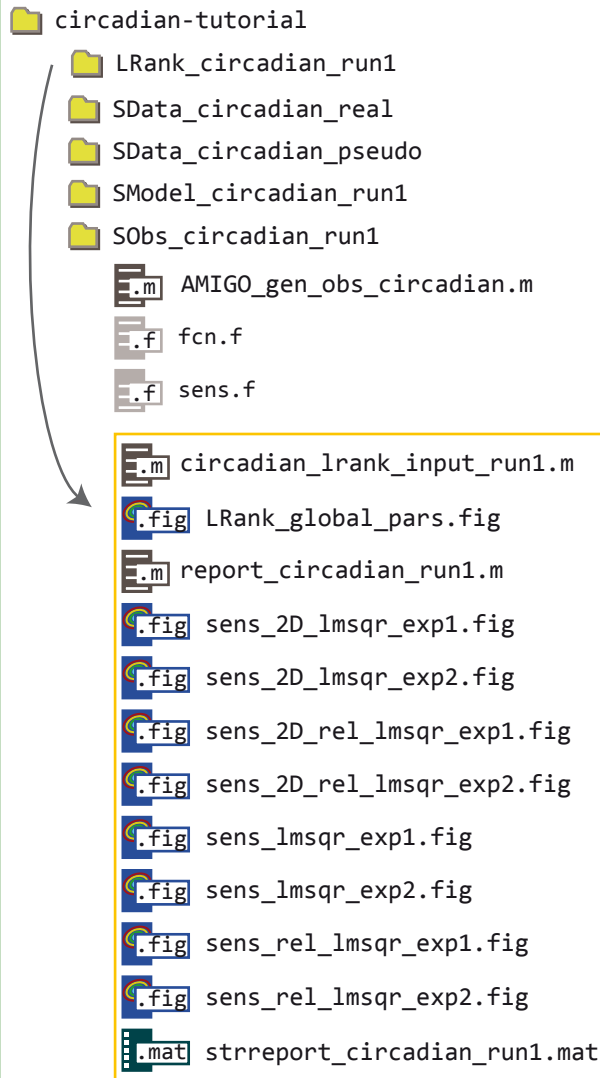
```
>> AMIGO_LRank('circadian_tutorial_lrank')
```

Rank can be performed for unknown global and local parameters and unknown global and local initial conditions. Results are (also) depicted per experiment .

Local Rank

(Minimum) Outputs

....\AMIGO\Results



Illustrative example: The circadian clock in Arabidopsis thaliana

```
>> load report_struct.mat
```

```
>> results
```

```
  pathd: [1x1 struct]
```

```
  plotd: [1x1 struct]
```

```
  sim: [1x1 struct]
```

```
  rank: [1x1 struct]
```

```
>> results.rank
```

```
    sens_t: {[100x2x27 double] [100x2x27 double]}
```

```
    r_sens_t: {[100x2x27 double] [100x2x27 double]}
```

```
    rank_mat: {[27x5 double] [27x5 double]}
```

```
    r_rank_mat: {[27x5 double] [27x5 double]}
```

```
    sorted_par_rank_mat: {[27x5 double] [27x5 double]}
```

```
    r_sorted_par_rank_mat: {[27x5 double] [27x5 double]}
```

```
    par_rank_index: {[27x1 double] [27x1 double]}
```

```
    r_par_rank_index: {[27x1 double] [27x1 double]}
```

```
    over_par_rank_index: [27x1 double]
```

```
    sorted_over_par_rank_mat: [27x5 double]
```

```
    r_over_par_rank_index: [27x1 double]
```

```
    r_sorted_over_par_rank_mat: [27x5 double]
```

```
    d_obs_par_msqr: {[2x27 double] [2x27 double]}
```

```
    d_obs_par_mabs: {[2x27 double] [2x27 double]}
```

```
    d_obs_par_mean: {[2x27 double] [2x27 double]}
```

```
    r_d_obs_par_msqr: {[2x27 double] [2x27 double]}
```

```
    r_d_obs_par_mabs: {[2x27 double] [2x27 double]}
```

```
    r_d_obs_par_mean: {[2x27 double] [2x27 double]}
```

```
    y0_rank_index: {[ ] [ ]}
```

```
    r_y0_rank_index: {[ ] [ ]}
```

```
    d_obs_y0_msqr: {[ ] [ ]}
```

```
    d_obs_y0_mabs: {[ ] [ ]}
```

```
    d_obs_y0_mean: {[ ] [ ]}
```

```
    r_d_obs_y0_msqr: {[ ] [ ]}
```

```
    r_d_obs_y0_mabs: {[ ] [ ]}
```

```
    r_d_obs_y0_mean: {[ ] [ ]}
```

```
    r_sorted_y0_rank_mat: {[0x5 double] [0x5 double]}
```

```
    sorted_y0_rank_mat: {[0x5 double] [0x5 double]}
```

```
    d_obs_msqr: {[2x27 double] [2x27 double]}
```

```
    d_obs_mabs: {[2x27 double] [2x27 double]}
```

```
    d_obs_mean: {[2x27 double] [2x27 double]}
```

```
    r_d_obs_msqr: {[2x27 double] [2x27 double]}
```

```
    r_d_obs_mabs: {[2x27 double] [2x27 double]}
```

```
    r_d_obs_mean: {[2x27 double] [2x27 double]}
```

Absolute & relative sensitivities vs time

Absolute & relative rank values for the unknown parameters for each experiment

Overall absolute & relative rank for the unknown parameters (summation over all experiments)

Absolute & relative rank values for the unknown parameters for each experiment for each observable

Absolute & relative rank values for the unknown initial conditions for each experiment

Overall absolute & relative rank for the unknown initial conditions for all experiments

Absolute & relative rank values for the unknowns for each experiment for each observable

 circadian_lrank_input_run1.m
 LRank_global_pars.fig
 report_circadian_run1.m
 sens_2D_lmsqr_exp1.fig
 sens_2D_lmsqr_exp2.fig
 sens_2D_rel_lmsqr_exp1.fig
 sens_2D_rel_lmsqr_exp2.fig
 sens_lmsqr_exp1.fig
 sens_lmsqr_exp2.fig
 sens_rel_lmsqr_exp1.fig
 sens_rel_lmsqr_exp2.fig
 strreport_circadian_run1.mat

-----> RANKING for experiment: 1

.....

----->RELATIVE Ranking of model unknowns:

	par	value	rd_msqr	rd_mabs	rd_mean	rd_max	rd_min
	n1	7.5038e+000	5.8197e-001	2.3975e+000	-2.1391e-001	5.3477e+000	-9.0290e+000
	m1	1.0098e+001	5.2957e-001	2.1709e+000	2.5133e-001	8.3699e+000	-4.5605e+000
	m4	2.3422e+000	4.9860e-001	2.2389e+000	2.2913e-001	5.7671e+000	-3.7213e+000
	p1	2.1994e+000	4.4056e-001	1.6193e+000	-6.0023e-001	3.1811e+000	-7.3332e+000
	n2	6.8010e-001	4.3181e-001	1.9627e+000	-1.5869e-001	3.8258e+000	-4.5262e+000
	m5	7.2482e+000	3.7960e-001	1.7224e+000	4.2848e-001	3.4327e+000	-3.1264e+000
	k1	3.8045e+000	3.6751e-001	1.5235e+000	-1.3764e-001	3.3939e+000	-5.6741e+000
	g2	3.0412e+000	3.6495e-001	1.4280e+000	2.5471e-001	5.5839e+000	-3.3911e+000
	k4	2.5356e+000	3.5968e-001	1.6274e+000	-1.1642e-001	2.6028e+000	-4.0879e+000
	p2	9.4440e+000	3.4170e-001	1.4903e+000	-5.4599e-001	2.7072e+000	-3.1883e+000
	r3	4.3640e-001	3.3447e-001	1.3904e+000	-5.1992e-002	3.4368e+000	-4.9341e+000
	g1	1.4992e+000	3.3392e-001	1.3898e+000	8.1224e-002	5.0026e+000	-3.3152e+000
	r1	2.8170e-001	3.2966e-001	1.3251e+000	-9.2396e-002	3.3573e+000	-4.7969e+000
	r4	7.3021e+000	3.1062e-001	1.2897e+000	5.9467e-002	4.6282e+000	-3.1427e+000
	r2	7.6760e-001	1.7444e-001	6.9839e-001	8.7811e-002	2.7114e+000	-1.6065e+000
	m3	3.7511e+000	1.4525e-001	5.1105e-001	3.4119e-001	2.4949e+000	-5.2870e-001
	k3	4.1946e+000	9.5499e-002	3.3327e-001	-1.9416e-001	5.4566e-001	-1.5403e+000
	m2	1.9685e+000	8.8991e-002	3.1650e-001	2.6841e-001	1.3975e+000	-1.9233e-001
	k2	5.3087e+000	4.6735e-002	1.8663e-001	-6.9922e-002	3.8532e-001	-6.0282e-001
	k5	1.4420e+000	4.0570e-002	1.4678e-001	2.9100e-002	5.1211e-001	-2.5300e-001
	m6	1.8981e+000	2.1757e-002	9.2196e-002	3.2929e-002	2.3163e-001	-1.3564e-001
	k6	4.8600e+000	1.7033e-002	7.2930e-002	-2.5749e-002	1.0768e-001	-1.7460e-001
	m7	1.2000e+000	0.0000e+000	0.0000e+000	0.0000e+000	0.0000e+000	0.0000e+000
	k7	1.2000e+000	0.0000e+000	0.0000e+000	0.0000e+000	0.0000e+000	0.0000e+000
	p3	5.0000e-001	0.0000e+000	0.0000e+000	0.0000e+000	0.0000e+000	0.0000e+000
	q1	4.5703e+000	0.0000e+000	0.0000e+000	0.0000e+000	0.0000e+000	0.0000e+000
	q2	1.0000e+000	0.0000e+000	0.0000e+000	0.0000e+000	0.0000e+000	0.0000e+000

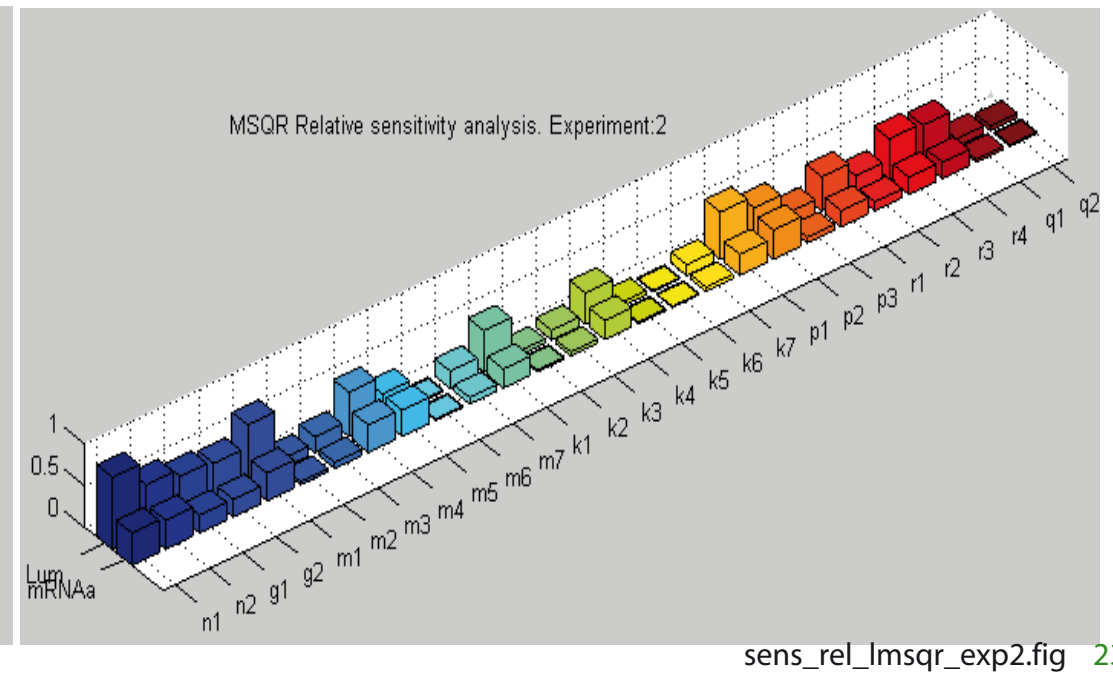
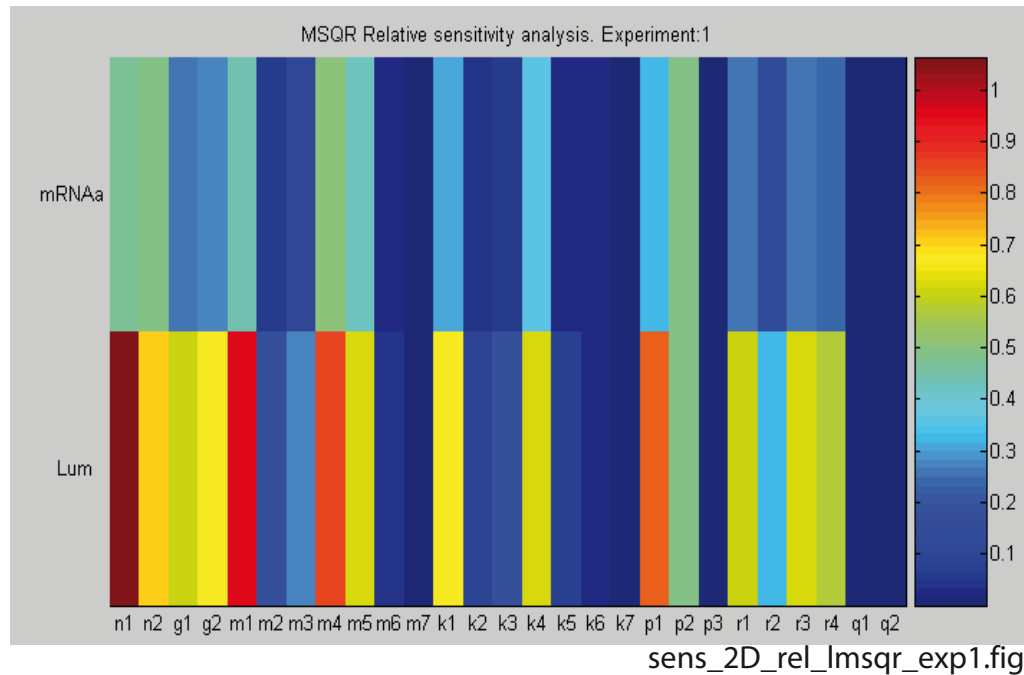
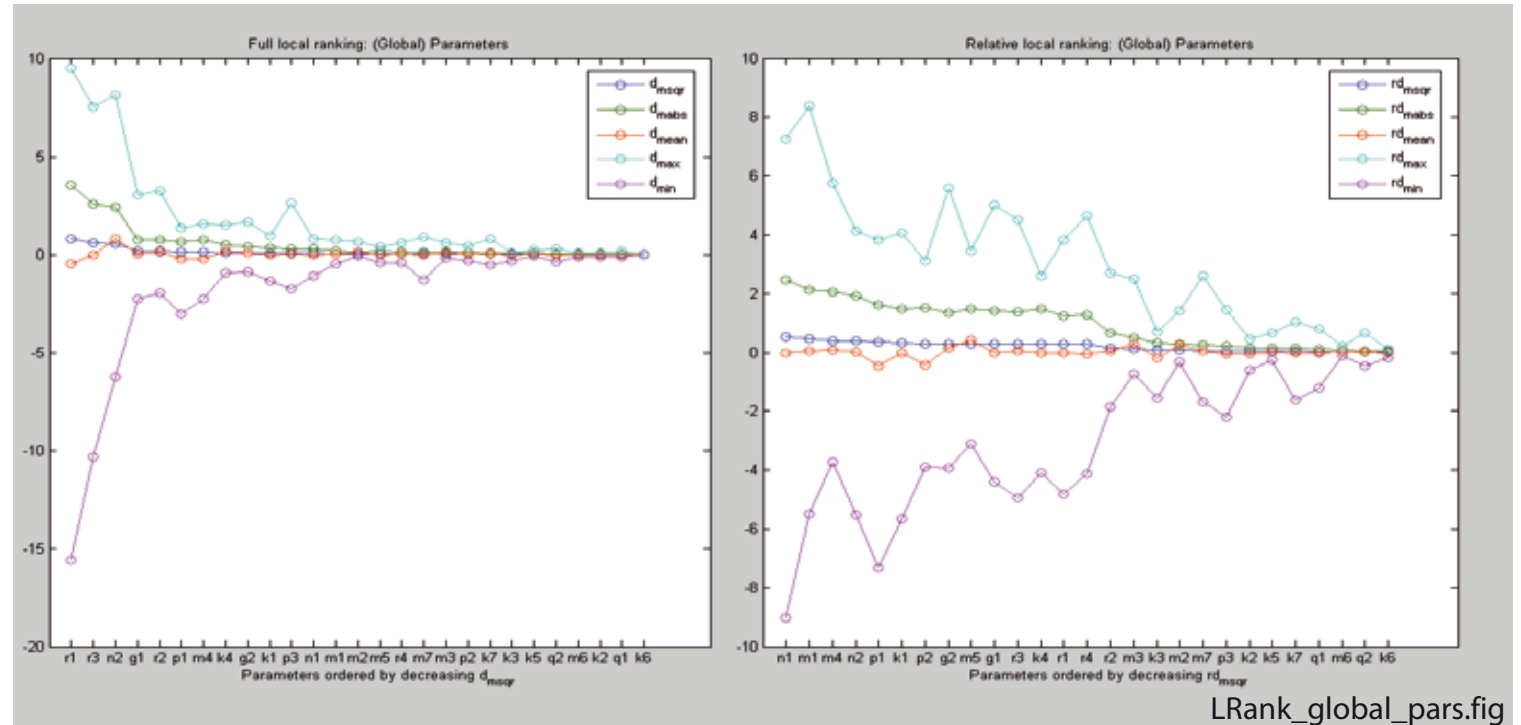
.....

Local Rank

Outputs

 circadian_lrank_input_run1.m
 LRank_global_pars.eps
 LRank_global_pars.fig
 report_circadian_run1.m
 sens_2D_lmsqr_exp1.fig
 sens_2D_lmsqr_exp2.fig
 sens_2D_rel_lmsqr_exp1.fig
 sens_2D_rel_lmsqr_exp2.fig
 sens_lmsqr_exp1.fig
 sens_lmsqr_exp2.fig
 sens_rel_lmsqr_exp1.fig
 sens_rel_lmsqr_exp2.fig
 strreport_circadian_run1.mat

Illustrative example: The circadian clock in *Arabidopsis thaliana*



Inputs

```
% PATHS RELATED DATA
results.pathd.results_folder='circadian-tutorial';
results.pathd.short_name='circadian';

% MODEL RELATED DATA
inputs.model.input_model_type='charmodelF';
inputs.model.n_st=7; inputs.model.n_par=27; inputs.model.n_stimulus=1;
inputs.model.st_names=char('CL_m','CL_c','CL_n','CT_m','CT_c','CT_n','CP_n');
inputs.model.par_names=char('n1','n2','g1','g2','m1','m2','m3','m4','m5',...
    'm6','m7','k1','k2','k3','k4','k5','k6','k7',...
    'p1','p2','p3','r1','r2','r3','r4','q1','q2');
inputs.model.stimulus_names=char('light');
inputs.model.eqns=...
char('dCL_m=q1*CP_n*light+n1*CT_n/(g1+CT_n)-m1*CL_m/(k1+CL_m)',...
    'dCL_c=p1*CL_m-r1*CL_c+r2*CL_n-m2*CL_c/(k2+CL_c)',...
    'dCL_n=r1*CL_c-r2*CL_n-m3*CL_n/(k3+CL_n)',...
    'dCT_m=n2*g2**2/(g2+CL_n**2)-m4*CT_m/(k4+CT_m)',...
    'dCT_c=p2*CT_m-r3*CT_c+r4*CT_n-m5*CT_c/(k5+CT_c)',...
    'dCT_n=r3*CT_c-r4*CT_n-m6*CT_n/(k6+CT_n)',...
    'dC_Pn=(1-light)*p3-m7*CP_n/(k7+CP_n)-q2*light*CP_n');
inputs.model.par=[7.5038 0.6801 1.4992 3.0412 10.0982 1.9685 3.7511 2.3422 ...
    7.2482 1.8981 1.2 3.8045 5.3087 4.1946 2.5356 1.4420 4.8600 ...
    1.2 2.1994 9.4440 0.5 0.2817 0.7676 0.4364 7.3021 4.5703 1.0];

% EXPERIMENTAL SCHEME RELATED DATA
inputs.exps.n_exp=2;
for iexp=1:inputs.exps.n_exp
inputs.exps.exp_y0{iexp}=zeros(1,inputs.model.n_st);inputs.exps.t_f{iexp}=120;
end
inputs.exps.u_interp{1}='sustained'; inputs.exps.t_con{1}=[0 120];
inputs.exps.u{1}=[1];
inputs.exps.u_interp{2}='pulse-down'; inputs.exps.n_pulses{2}=5;
inputs.exps.t_con{2}=[0 :12: 120];

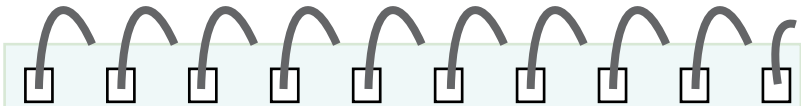
% EXPERIMENTAL DATA RELATED INFO
inputs.exps.n_s{1}=15; inputs.exps.n_s{2}=25;

% UNKNOWNNS RELATED DATA

% To rank all parameters:
inputs.PEsol.id_global_theta='all';
inputs.PEsol.global_theta_max=20.*ones(1,inputs.model.n_par);
inputs.PEsol.global_theta_min=zeros(1,inputs.model.n_par);
```

Illustrative example: The circadian clock in Arabidopsis thaliana

Global rank of parameters under two experiments:
1st under sustained stimulation with 15 equidistant sampling times
2nd under pulse-wise stimulation with 25 equidistant sampling times



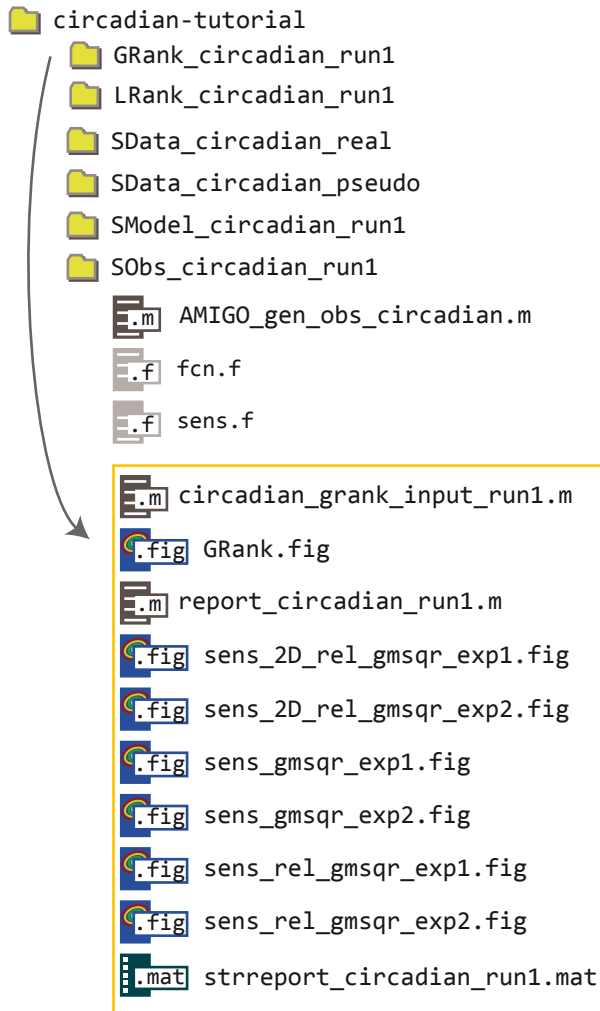
How to rank the parameters ?

```
>> AMIGO_GRank('circadian_grank')
```

Global Rank

(Minimum) Outputs

....\AMIGO\Results



Illustrative example: The circadian clock in Arabidopsis thaliana

```
>> load report_struct.mat
```

```
>> results
```

```
pathd: [1x1 struct]
```

```
plotd: [1x1 struct]
```

```
rank: [1x1 struct]
```

```
>> results.rank
```

```
n_global_samples: 10001
```

```
global_par_rank_index: [27x1 double]
```

```
r_global_par_rank_index: [27x1 double]
```

```
global_par_rank_mat: [27x5 double]
```

```
r_global_par_rank_mat: [27x5 double]
```

```
global_y0_rank_index: [0x1 double]
```

```
r_global_y0_rank_index: [0x1 double]
```

```
global_y0_rank_mat: [0x5 double]
```

```
r_global_y0_rank_mat: [0x5 double]
```

```
g_d_obs_msqr_mat: {[2x27 double] [2x27 double]}
```

```
g_d_obs_mabs_mat: {[2x27 double] [2x27 double]}
```

```
g_d_obs_mean_mat: {[2x27 double] [2x27 double]}
```

```
g_r_d_obs_msqr_mat: {[2x27 double] [2x27 double]}
```

```
g_r_d_obs_mabs_mat: {[2x27 double] [2x27 double]}
```

```
g_r_d_obs_mean_mat: {[2x27 double] [2x27 double]}
```

Number of samples within bounds

Absolute & relative global rank values for the unknown parameters

Absolute & relative global rank values for the unknown initial conditions

Absolute & relative rank values for the unknowns for each experiment for each observable

Fragment of report_circadian_run1.m

```
-----> GLOBAL RANKING
```

```
-----> ABSOLUTE Ranking of GLOBAL unknown PARAMETERS:
```

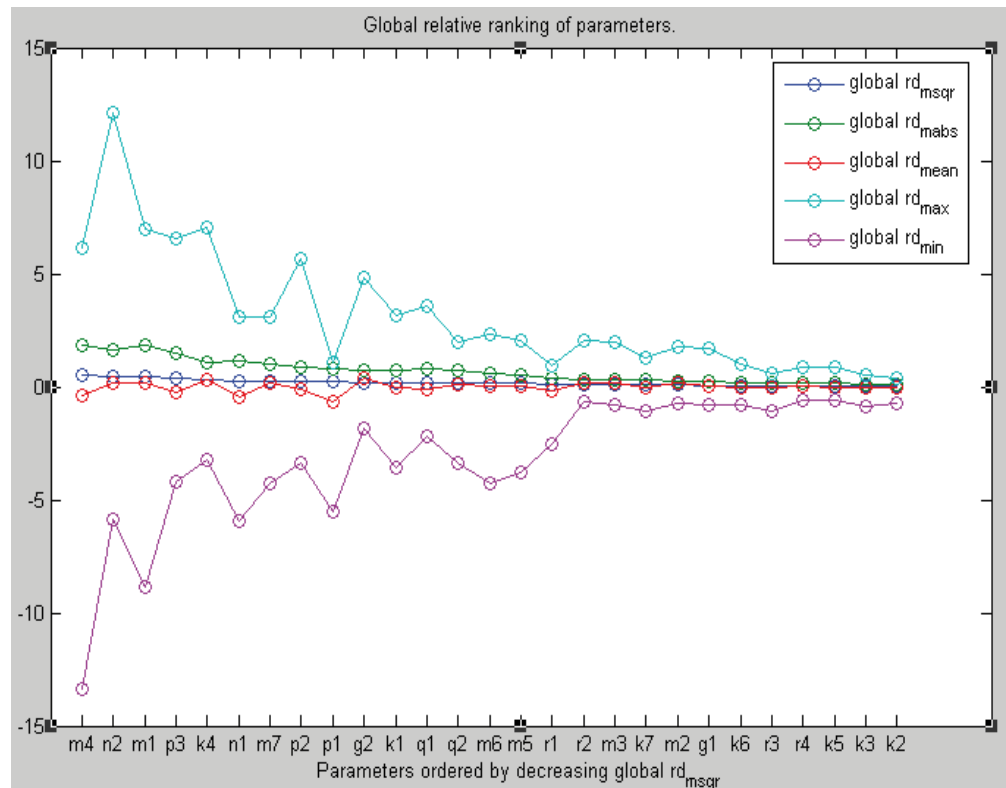
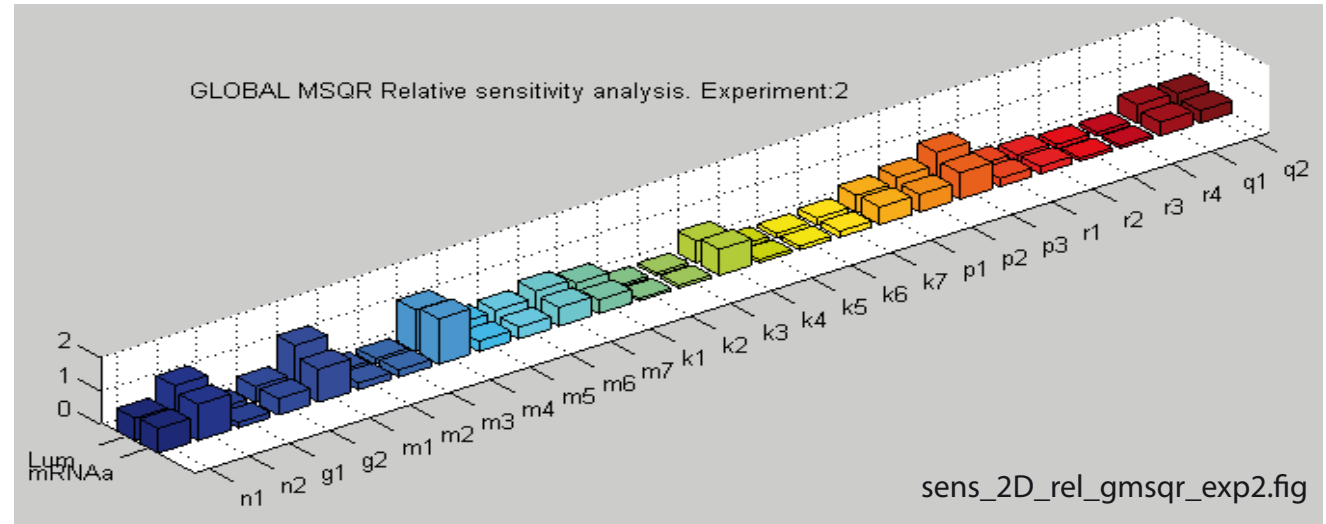
	d_msqr	d_mabs	d_mean	d_max	d_min
q2	4.9385e+002	1.4355e+002	-1.4336e+002	7.5474e-001	-7.6671e+002
p3	2.2486e+001	7.1256e+000	6.6883e+000	2.9918e+001	-1.4414e+000
m7	2.2386e+001	7.0398e+000	-6.8126e+000	7.8047e-001	-3.0582e+001
q1	1.7648e+001	5.5747e+000	5.2387e+000	2.3476e+001	-1.3400e+000
p1	1.6703e+001	4.6149e+000	-4.4917e+000	9.4374e-001	-2.7898e+001
r1	1.5860e+001	4.1703e+000	-4.1339e+000	3.3310e-001	-2.7248e+001
m4	1.3782e+001	3.8194e+000	-3.6880e+000	1.3159e+000	-2.1965e+001
m1	1.3146e+001	4.2694e+000	-3.5583e+000	2.5954e+000	-1.7984e+001
n1	1.0464e+001	3.4722e+000	7.0321e-001	9.3716e+000	-7.7203e+000
n2	7.8374e+000	2.2438e+000	2.0351e+000	1.2299e+001	-2.0747e+000

Global Rank

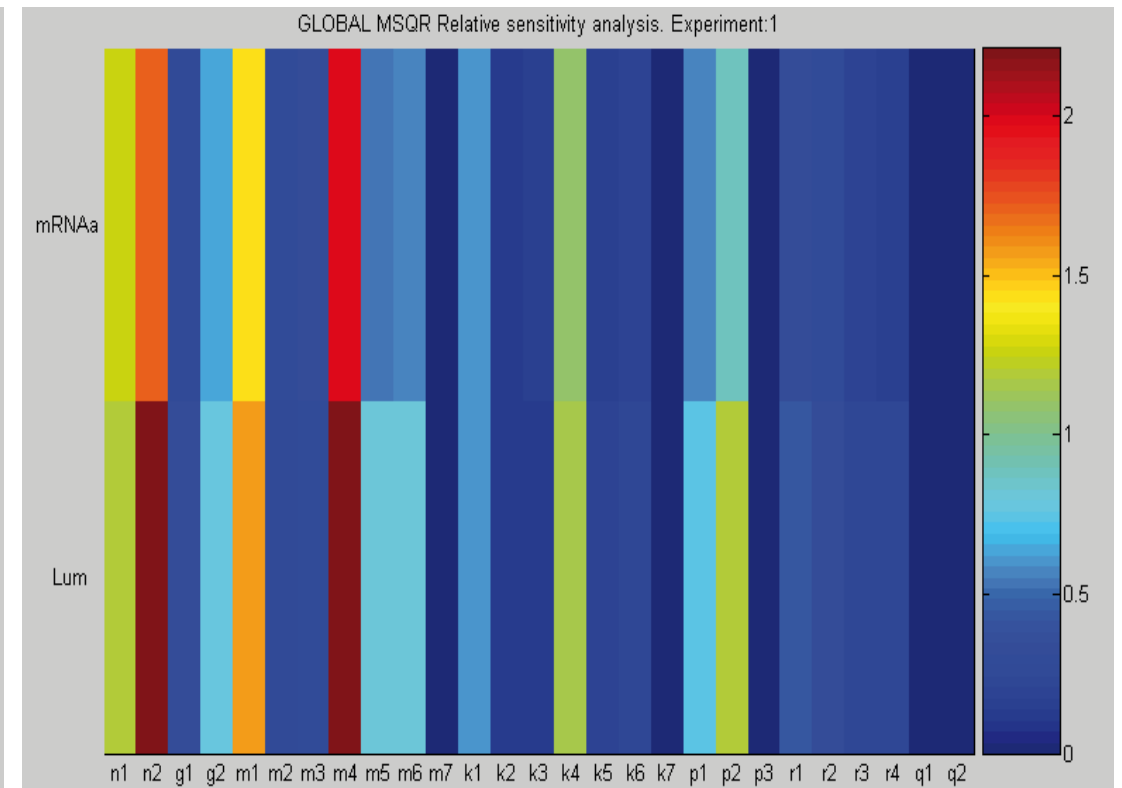
Outputs

- strreport_circadian_run1.mat
- sens_rel_gmsqr_exp2.fig
- sens_rel_gmsqr_exp1.fig
- sens_gmsqr_exp2.fig
- sens_gmsqr_exp1.fig
- sens_2D_rel_gmsqr_exp2.fig
- sens_2D_rel_gmsqr_exp1.fig
- report_circadian_run1.m
- GRank.fig
- circadian_grank_input_run1.m

Illustrative example: The circadian clock in *Arabidopsis thaliana*



GRank.fig



sens_rel_gmsqr_exp1.fig

Parameter estimation

> *AMIGO_PE*

Estimates global and local (experiment dependent) model unknowns by using global and/or local optimization methods.

Inputs

Models

- > ***Deterministic ODEs*** Any non-linear general form
- > ***Format*** May be provided in MATLAB, FORTRAN, SBML, strings, black-box
- > ***Observation functions*** Any linear / non-linear function of the states
- > ***Notation*** Customized names for variables, states & observables are allowed

Experimental Scheme

- > ***Experiments info*** Flexible number of experiments
Initial conditions [per experiment, for rank per experiments]
Experiment duration [per experiment]
Unknown values [per experiment, for rank per experiments]
Observables [per experiment]
[Stimuli conditions per experiment: sustained, step-wise, pulse-wise, linear interpolation]

IVP Solvers

- > ***Explicit*** Runge-Kutta type methods: RKF45; Adams method: ode113 Matlab
- > ***Implicit*** Methods for stiff systems: Radau5, ode15s Matlab, LSODA, LSODES
- > ***Programming*** Mex-files to FORTRAN compiled code are used to accelerate simulation

NLP Solvers

- > ***Local methods*** fmincon, fsqp, ipopt, nomad, dhc, n2fb, dn2fb, fminsearch, ...
- > ***Multistart of local***
- > ***Global /hybrid*** SSm, DE, SRES, sequential hybrids: DE+local and SRES+local

Outputs

- > ***Report*** .m file that keeps input data and best unknowns with confidence intervals
- > ***Structure*** .mat that keeps inputs. and results. pathd,.plotd,.nlpsol,.fit,.sim structures
- > ***Figures*** .fig [and .eps] files with the best fits, residuals, correlation matrix (when available in the optimum), convergence curve, statistics of results for multistart

Parameter estimation

Inputs

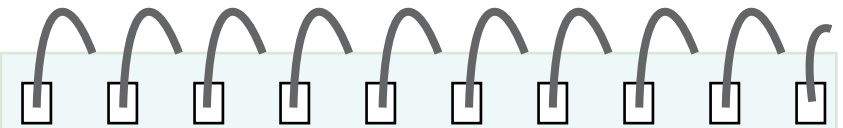
```
% PATHS RELATED DATA
%.... as in previous slides
% MODEL RELATED DATA
%.... as in previous slides
% EXPERIMENTAL SCHEME RELATED DATA
%.... as in previous slides
% EXPERIMENTAL DATA RELATED INFO
inputs.exps.data_type='real';
inputs.exps.noise_type='homo_var';
inputs.exps.n_s{1}=15;
inputs.exps.n_s{2}=25;
inputs.exps.exp_data{1}= [0.037642 0.059832
                           1.398618 0.983442
                           .....
                           1.389312 0.678665
                           0.833228 0.574736];
inputs.exps.error_data{1}=[0.037642 0.059832
                           0.072461 0.013999
                           .....
                           0.054665 0.066953
                           0.082163 0.015295];
inputs.exps.exp_data{2}= [0.146016 0.018152
                           0.831813 1.002499
                           .....
                           1.622378 0.824711
                           1.525194 0.537398];
inputs.exps.error_data{2}=[0.146016 0.018152
                           0.066547 0.045194
                           .....
                           0.099239 0.002678
                           0.010644 0.052990];

% UNKNOWNNS RELATED DATA
% To estimate GLOBAL selected parameters
inputs.PEsol.id_global_theta=...
char('n1;n2;m1;m4;m6;m7;k1;k4;p3;q1');
inputs.PEsol.global_theta_max=[15 15 15 15 15 15 15 15 15];
inputs.PEsol.global_theta_min=[0 0 0 0 0 0 0 0 0];
```



Illustrative example: The circadian clock in Arabidopsis thaliana

Parameter estimation using two sets of experiments subject to gaussian homoscedastic experimental noise.
1st under sustained stimulation with 15 equidistant sampling times and 2nd under pulse-wise stimulation with 25 equidistant sampling times



How to estimate the parameters ?

```
% With the multistart of ...

>> AMIGO_PE('circadian_pe',run1,'multi_fmincon')

% With the default solver (SSm)

>> AMIGO_PE('circadian_pe')

% Other examples

>> AMIGO_PE('circadian_pe',run1,'de')
>> AMIGO_PE('circadian_pe',run2,'de')
>> AMIGO_PE('circadian_pe',p1,'hyb_de_fmincon')
```

NLP solver may be indicated in the input file or in command line

All solvers have options that may be modified in the corresponding solver-name_options.m file



Parameter estimation

(Minimum) Outputs

....\AMIGO\Results

circadian-tutorial

- GRank_circadian_run1
- LRank_circadian_run1
- PE_circadian_de_run1
- PE_circadian_de_run2
- PE_circadian_hyb_de_dn2fb_p1
- PE_circadian_hyb_de_fmincon_p1
- PE_circadian_multi_fmincon_run1
- PE_circadian_ssm_run1
- SData_circadian_real
- SData_circadian_pseudo
- SModel_circadian_run1
- SObs_circadian_run1
- AMIGO_gen_obs_circadian.m
- fcn.f
- sens.f

- circadian_pe_input_run1.m
- conv_curvefig
- corr_mat.fig
- fit_plot_exp1.fig
- fit_plot_exp2.fig
- report_circadian_run1.m
- residuals_meanmax.fig
- strreport_circadian_run1.mat

Illustrative example: The circadian clock in Arabidopsis thaliana

**SOLUTION WITH
MULTISTART OF A LOCAL**

```
>> results
pathd: [1x1 struct]
plotd: [1x1 struct]
nlpsol: [1x1 struct]
fit: [1x1 struct]
sim: [1x1 struct]
```

```
>> results.nlpsol
```

```
fbest: 36.3547
vbest: [6.1306 0.7094 8.9541 2.3850 2.1020 1.2903 4.3052 2.4560 0.4903]
cpu_time: 30.0313
conv_curve: [7x2 double]
```

Statistics and solution of the
NLP problem

```
>> results.fit
```

```
residuals: {[15x2 double] [25x2 double]}
rel_residuals: {[15x2 double] [25x2 double]}
ms: {[15x2 double] [25x2 double]}
g_FIM: [9x9 double]
g_corr_mat: [10x10 double]
g_var_cov_mat: [9x9 double]
thetabest: [6.1306 0.7094 8.9541 2.3850 2.1020 1.2903 4.3052 2.4560 0.4903]
fbest: 36.3547
cpu_time: 30.0313
```

Relative and absolute residuals
Model prediction for the optimal
value of unknowns

Global FIM, Covariance Matrix and
Correlation matrix

Optimal solution and overall
computational cost

```
>>> Estimated global parameters:
```

```
n1: 6.1306e+000 +- 1.6186e+000;
n2: 7.0943e-001 +- 4.0134e-002;
m1: 8.9541e+000 +- 1.4441e+000;
```

Fragment of
report_circadian_run1.m

```
.....
>>> Correlation matrix for the global unknowns:
```

```
1.000000e+000 -9.284903e-001 8.173269e-001 -3.189126e-002 ....
-9.284903e-001 1.000000e+000 -6.606572e-001 .....
```

```
>>>> Mean / Maximum value of the residuals in percentage:
```

Experiment 1 :

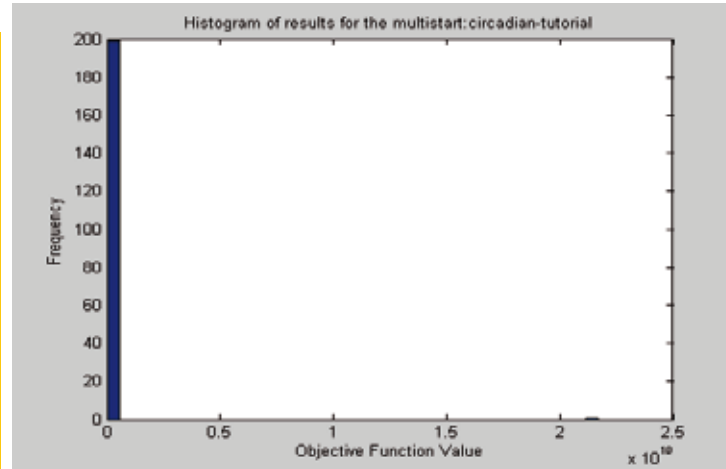
Observable 1 --> mean error: 13.234282 % max error: 100.000000 %.....

Parameter estimation

(Minimum) Outputs

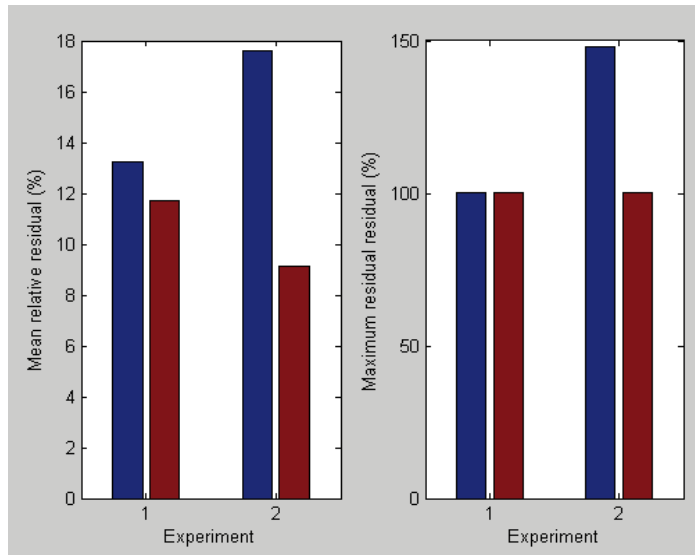
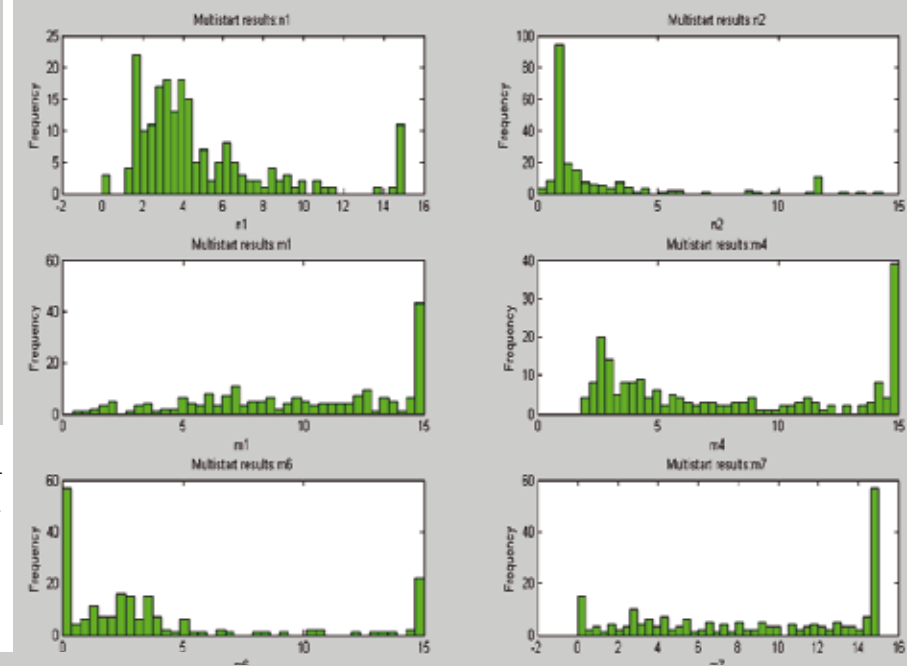
```

.m circadian_pe_input_run1.m
.fig conv_curvefig
.fig corr_mat.fig
.fig fit_plot_exp1.fig
.fig fit_plot_exp2.fig
.m report_circadian_run1.m
.fig residuals_meanmax.fig
.mat strreport_circadian_run1.mat
    
```

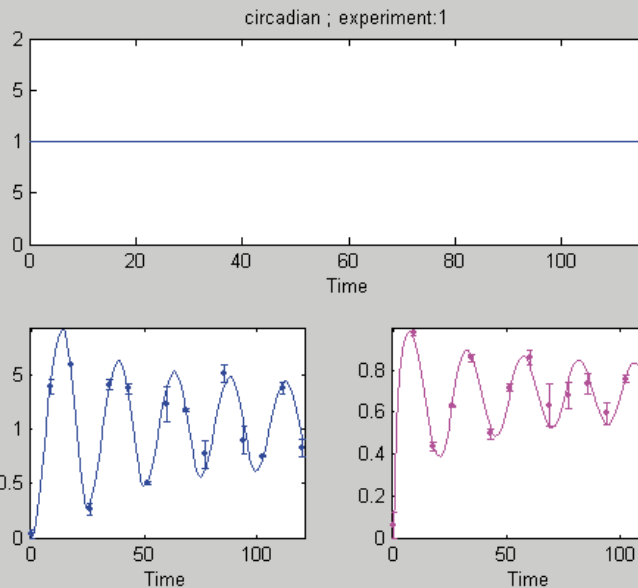


Histogram of solutions achieved in the multistart

Illustrative example: The circadian clock in *Arabidopsis thaliana*

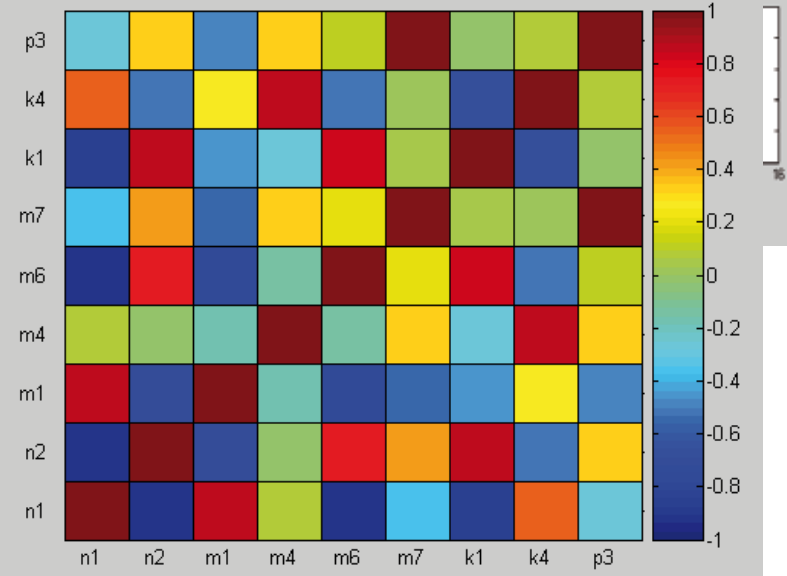


Relative mean and max residuals



Best fit

Crammer Rao based correlation matrix for global unknowns



Correlation matrix for the solution found

Identifiability analysis: Cost contours

> **AMIGO_ContourP** Plots log-likelihood or least squares plots for each parameter and contour plots by pairs of parameters within the given bounds

Inputs	Models	> Deterministic ODEs > Format > Observation functions > Notation	Any non-linear general form May be provided in MATLAB, FORTRAN, SBML, strings, black-box Any linear / non-linear function of the states Customized names for variables, states & observables are allowed
	Experimental Data	> Experimental Scheme > Real data > Experimental error info	Flexible number of experiments; Initial conditions [per experiment] Experiment duration [per experiment]; Unknown values [per experiment] Observables [per experiment] ; [Stimuli conditions per experiment: sustained, step-wise, pulse-wise, linear interpolation]; Sampling times [per experiment] Matrices of experimental data and error bars if available Normal error assumed: homoscedastic (constant or variable variance) heteroscedastic (power in the mean)
	IVP Solvers	> Explicit > Implicit > Programming	Runge-Kutta type methods: RKF45; Adams method: ode113 Matlab Methods for stiff systems: Radau5, ode15s Matlab, LSODA, LSODES Mex-files to FORTRAN compiled code are used to accelerate simulation
	Outputs	> Report > Structure > Figures	.m file that keeps input data .mat that keeps inputs. and results.pathd, .plotted, .contour structures .fig [and .eps] files with 1D or 2D contour plots by pairs of parameters

Identifiability analysis: Cost contours

Inputs

```
% PATHS RELATED DATA
%.... as in previous slides
% MODEL RELATED DATA
%.... as in previous slides
% EXPERIMENTAL SCHEME RELATED DATA
%.... as in previous slides
% EXPERIMENTAL DATA RELATED INFO
inputs.exps.data_type='real';
inputs.exps.noise_type='homo_var';
inputs.exps.n_s{1}=15;
inputs.exps.n_s{2}=25;
inputs.exps.exp_data{1}= [0.037642 0.059832
                          1.398618 0.983442
                          .....
                          1.389312 0.678665
                          0.833228 0.574736];
inputs.exps.error_data{1}=[0.037642 0.059832
                           0.072461 0.013999
                           .....
                           0.054665 0.066953
                           0.082163 0.015295];
inputs.exps.exp_data{2}= [0.146016 0.018152
                          0.831813 1.002499
                          .....
                          1.622378 0.824711
                          1.525194 0.537398];
inputs.exps.error_data{2}=[0.146016 0.018152
                           0.066547 0.045194
                           .....
                           0.099239 0.002678
                           0.010644 0.052990];
```

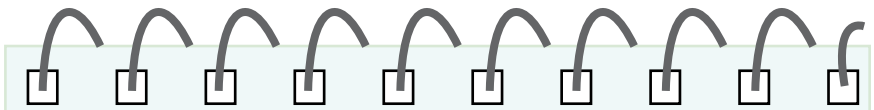
```
% UNKNOWNNS RELATED DATA
```

```
% To estimate GLOBAL selected parameters
```

```
inputs.PEsol.id_global_theta=...
char('n1","n2","m1","m4","m6","m7","k1","k4","p3","q1');
inputs.PEsol.global_theta_guess=[6.1307 7.0943e-001 8.9540 2.3850 2.1019 ...
                                1.2904 4.305 2.4561 4.9036e-001];
inputs.PEsol.global_theta_max=1.25.*inputs.PEsol.global_theta_guess;
inputs.PEsol.global_theta_min=0.75.*inputs.PEsol.global_theta_guess;
```

Illustrative example: The circadian clock in Arabidopsis thaliana

Contours of the log-likelihood function in the vicinity of the global solution for the model unknowns and under the two experiments scheme.



How to plot cost contours ?

```
>> AMIGO_ContourP('circadian_ident')
```

Identifiability analysis: Cost contours

(Minimum) Outputs

....\AMIGO\Results

circadian-tutorial

- Contours_circadian_run1
- GRank_circadian_run1
- LRank_circadian_run1
- PE_circadian_de_run1
- PE_circadian_de_run2
- PE_circadian_hyb_de_dn2fb_p1
- PE_circadian_hyb_de_fmincon_p1
- PE_circadian_multi_fmincon_run1
- PE_circadian_ssm_run1
- SData_circadian_real
- SData_circadian_pseudo
- SModel_circadian_run1
- SObs_circadian_run1
- AMIGO_gen_obs_circadian.m
- fcn.f
- sens.f

- circadian_ident_input_run1.m
- contourP_k1_vs_m1.fig
- contourP_k1_vs_m4.fig
- contourP_k1_vs_m6.fig
- contourP_k1_vs_m7.fig
- contourP_k1_vs_n1.fig
- contourP_k1_vs_n2.fig
- contourP_k4_vs_k1.fig
- contourP_k4_vs_m1.fig

```
>> load strreport_circadian_run1.mat

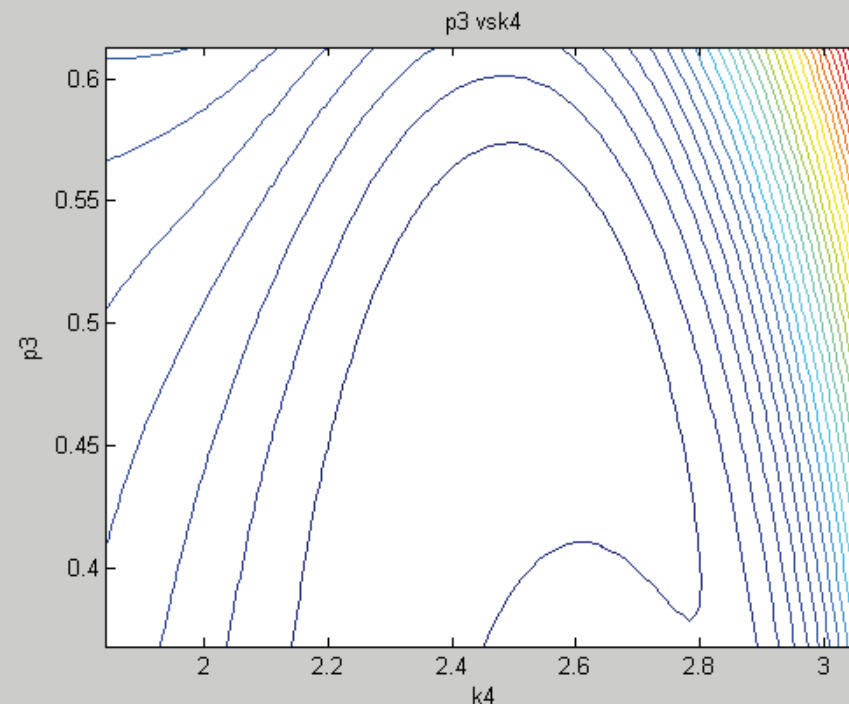
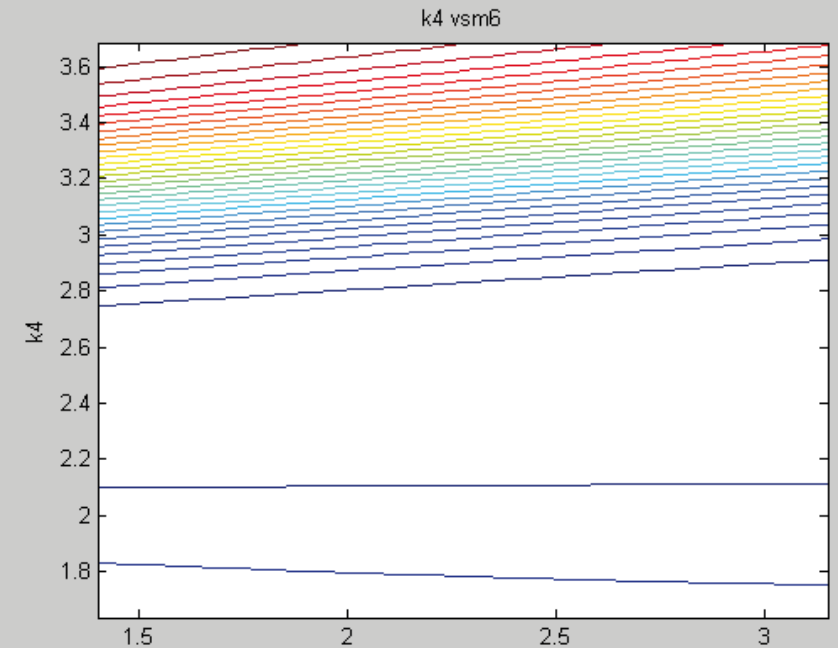
>> results
pathd: [1x1 struct]
plotd: [1x1 struct]
contour: [1x1 struct]

>> results.contour

error1d: {1x9 cell}
par1d: {1x9 cell}
errorxy: {8x9 cell}
parx: {1x8 cell}
pary: {1x9 cell}
```

Data to
generate
plots

Illustrative example: The circadian clock in *Arabidopsis thaliana*



Robust identifiability analysis

> **AMIGO_RIdent** Performs the Monte-Carlo based identifiability analysis. Computes robust confidence regions.

Inputs	Models	> Deterministic ODEs > Format > Observation functions > Notation	Any non-linear general form May be provided in MATLAB, FORTRAN, SBML, strings, black-box Any linear / non-linear function of the states Customized names for variables, states & observables are allowed
	Experimental Data	> Experimental Scheme > Real data > Experimental error info	Flexible number of experiments; Initial conditions [per experiment] Experiment duration [per experiment]; Unknown values [per experiment] Observables [per experiment] ; [Stimuli conditions per experiment: sustained, step-wise, pulse-wise, linear interpolation]; Sampling times [per experiment] Matrices of experimental data and error bars if available Normal error assumed: homoscedastic (constant or variable variance) heteroscedastic (power in the mean)
	IVP Solvers	> Explicit > Implicit > Programming	Runge-Kutta type methods: RKF45; Adams method: ode113 Matlab Methods for stiff systems: Radau5, ode15s Matlab, LSODA, LSODES Mex-files to FORTRAN compiled code are used to accelerate simulation
	NLP Solvers	> Local methods > Multistart of local > Global /hybrid	fmincon, fsqp, ipopt, nomad, dhc, n2fb, dn2fb, fminsearch, ... SSm, DE, SRES, sequential hybrids: DE+local and SRES+local
	Outputs	> Report > Structure > Figures	.m file that keeps input data and best unknowns with confidence intervals .mat that keeps inputs. and results.pahd, .plotd, .rid structures .fig [and .eps] files with the robust confidence intervals, cloud plots by pairs of parameters, and eccentricity plot

Robust identifiability analysis

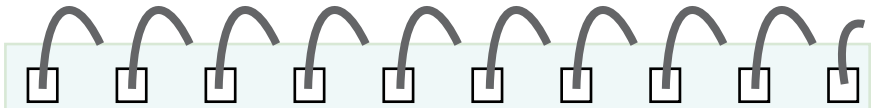
Inputs

```
% PATHS RELATED DATA
%.... as in previous slides
% MODEL RELATED DATA
%.... as in previous slides
% EXPERIMENTAL SCHEME RELATED DATA
%.... as in previous slides
% EXPERIMENTAL DATA RELATED INFO
inputs.exps.data_type='real';
inputs.exps.noise_type='homo_var';
inputs.exps.n_s{1}=15;
inputs.exps.n_s{2}=25;
inputs.exps.exp_data{1}= [0.037642 0.059832
                          1.398618 0.983442
                          .....
                          1.389312 0.678665
                          0.833228 0.574736];
inputs.exps.error_data{1}=[0.037642 0.059832
                           0.072461 0.013999
                           .....
                           0.054665 0.066953
                           0.082163 0.015295];
inputs.exps.exp_data{2}= [0.146016 0.018152
                          0.831813 1.002499
                          .....
                          1.622378 0.824711
                          1.525194 0.537398];
inputs.exps.error_data{2}=[0.146016 0.018152
                           0.066547 0.045194
                           .....
                           0.099239 0.002678
                           0.010644 0.052990];
```

```
% UNKNOWNNS RELATED DATA
% To estimate GLOBAL selected parameters
inputs.PEsol.id_global_theta=...
char('n1","n2","m1","m4","m6","m7","k1","k4","p3","q1');
inputs.PEsol.global_theta_guess=[6.1307 7.0943e-001 8.9540 2.3850 2.1019 ...
                                1.2904 4.305 2.4561 4.9036e-001];
inputs.PEsol.global_theta_max=1.25.*inputs.PEsol.global_theta_guess;
inputs.PEsol.global_theta_min=0.75.*inputs.PEsol.global_theta_guess;
```

Illustrative example: The circadian clock in Arabidopsis thaliana

Robust identifiability analysis of the global solution for the model unknowns and under the two experiments scheme.



How to perform robust identifiability analysis?

```
% From AMIGO path....
```

```
>> AMIGO_RIdent('circadian_ident')
```

Robust identifiability analysis

Outputs

circadian-tutorial

- Contours_circadian_run1
- GRank_circadian_run1
- LRank_circadian_run1
- PE_circadian_de_run1
- PE_circadian_de_run2
- PE_circadian_hyb_de_dn2fb_p1
- PE_circadian_hyb_de_fmincon_p1
- PE_circadian_multi_fmincon_run1
- PE_circadian_ssm_run1
- RIdent_circadian_ssm_run1
- SData_circadian_real
- SData_circadian_pseudo
- SModel_circadian_run1
- SObs_circadian_run1

.....

-  ccloud_n2_vs_m4.fig
-  ccloud_n2_vs_m6.fig
-  ccloud_n2_vs_m7.fig
-  ccloud_n2_vs_p3.fig
-  circadian_ident_input_run1.m
-  conf_k1.fig
-  conf_k4.fig
-  conf_m1.fig
-  conf_m4.fig

.....

Illustrative example: The circadian clock in *Arabidopsis thaliana*

```
>> load strreport_circadian_run1.mat
```

```
>> results
```

```
results =
```

```
  pathd: [1x1 struct]
```

```
  plotd: [1x1 struct]
```

```
  rid: [1x1 struct]
```

```
>> >> results.rid
```

```
    ecc: [9x9 double]
```

```
    alfa: [9x9 double]
```

```
  vtheta_guess: [6.1307 8.9540 2.3850 2.1019 1.2904 4.3050 2.4561 0.4904]
```

```
  sorted_dist: [1x500 double]
```

```
  sorted_dist_max: [1x500 double]
```

```
  sorted_dist_max95: [451x1 double]
```

```
  sorted_dist95: [451x1 double]
```

```
  sort_index_95: [1x450 double]
```

```
  best95: [450x9 double]
```

```
    mu: [7.6623 0.6796 10.2884 2.3455 1.8952 1.2135 3.8282 2.5453 0.5147]
```

```
  lambda: [1.5316 0.0298 1.3344 0.0395 0.2067 0.0769 0.4768 0.0892 0.0244]
```

```
  best95_norm: [450x9 double]
```

```
  confidence_interval: [1.5337 0.0596 2.1679 0.3077 0.6176 0.6662 1.8453 0.4778 0.2208]
```

```
  confidence_norm: [0.2002 0.0877 0.2107 0.1312 0.3259 0.5490 0.4820 0.1877 0.4290]
```

```
  semi_major: [8x9 double]
```

```
  semi_minor: [8x9 double]
```

```
    ecc_max: 12.5418
```

```
    ecc_min: 1.9859
```

```
    ecc_mean: 3.9707
```

```
  alfa_max: 3.1387
```

```
  alfa_min: 1.5046
```

```
  alfa_mean: 2.2807
```

```
  ellipse_pseudo_vol: 6.7482e-004
```

```
  lambda_total: 2.1008
```

```
  mc_corrmat: [10x10 double]
```

Information about eccentricity and angles of the confidence ellipses by pairs of parameters.

Information about the distance from results to the guess (global optimum). Information about the 95% confidence region.

Mean value of the unknowns and distance to the best (used as guess).

Confidence intervals: absolute and in tant per one.

Major and minor semiaxes for the confidence ellipses by pairs of parameters.

Information about eccentricity, angles and volume of the confidence hyperellipsoid

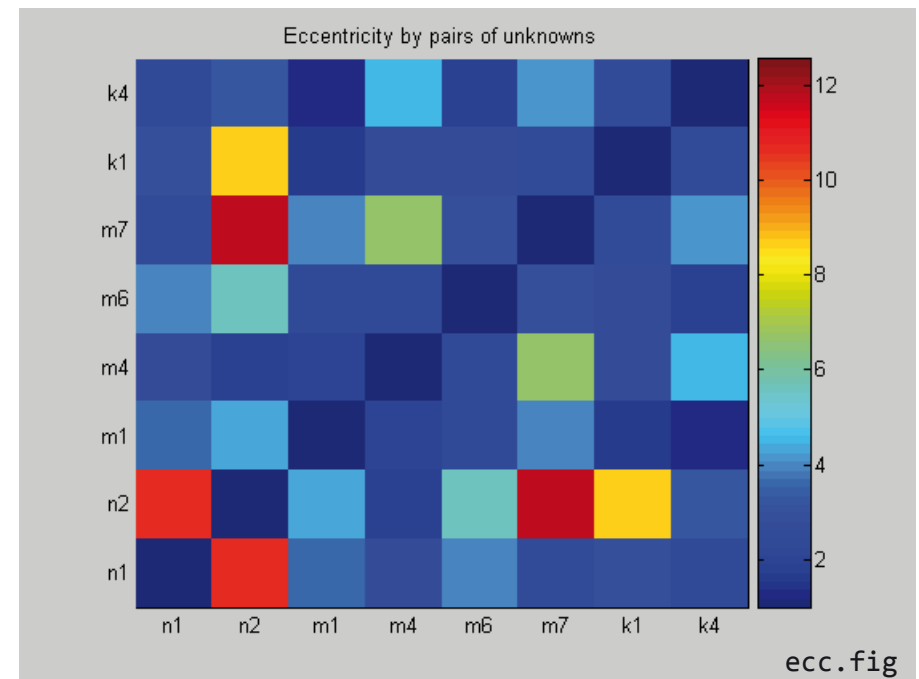
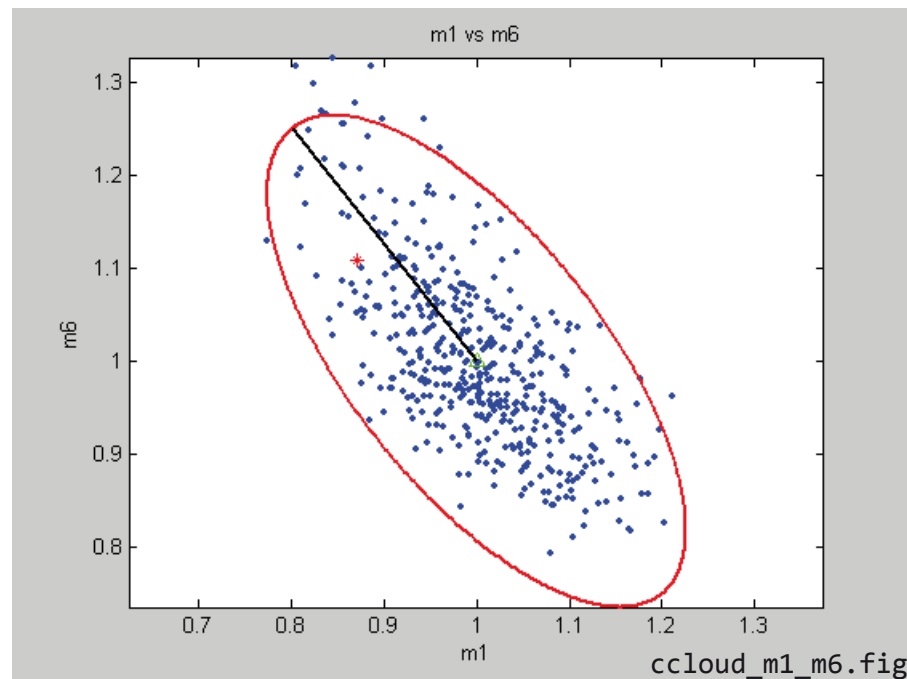
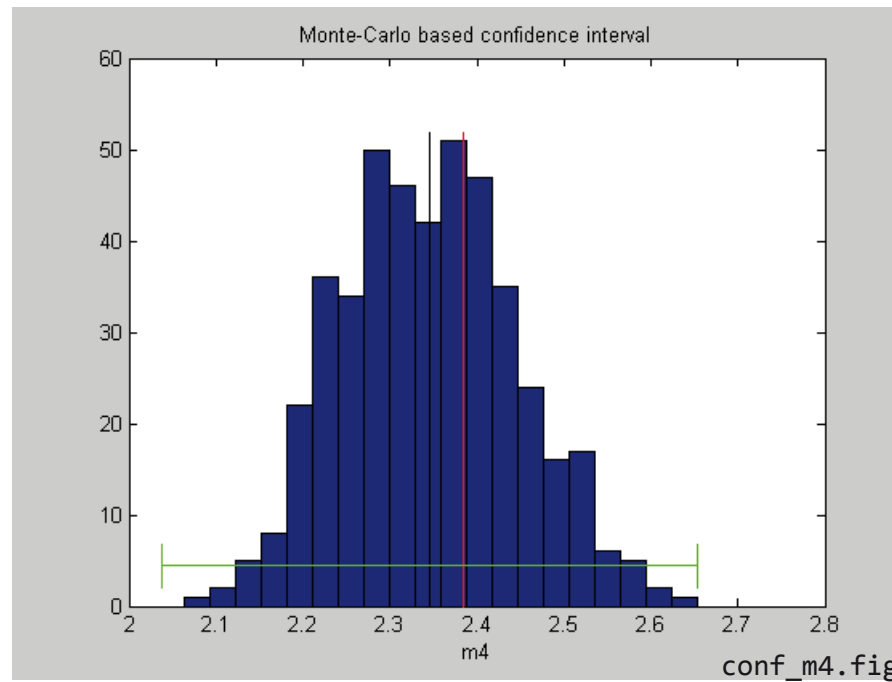
Monte-Carlo based correlation matrix

Robust identifiability analysis

Outputs

```
.....  
[.fig] ccloud_n2_vs_m4.fig  
[.fig] ccloud_n2_vs_m6.fig  
[.fig] ccloud_n2_vs_m7.fig  
[.fig] ccloud_n2_vs_p3.fig  
[.m] circadian_ident_input_run1.m  
[.fig] conf_k1.fig  
[.fig] conf_k4.fig  
[.fig] conf_m1.fig  
[.fig] conf_m4.fig  
.....
```

Illustrative example: The circadian clock in Arabidopsis thaliana



Optimal experimental design

> **AMIGO_OED** Performs FIM based optimal experimental design for the purpose of parameter estimation

Inputs	Models	> Deterministic ODEs > Format > Observation functions > Notation	Any non-linear general form May be provided in MATLAB, FORTRAN, SBML, strings, black-box Any linear / non-linear function of the states Customized names for variables, states & observables are allowed
	Experimental Scheme	> Experimental Scheme > Experimental error info	Flexible number of experiments; Initial conditions [per experiment] Experiment duration [per experiment]; Unknown values [per experiment] Observables [per experiment] ; [Stimuli conditions per experiment: sustained, step-wise, pulse-wise, linear interpolation]; Sampling times [per experiment] Normal error assumed: homoscedastic (constant or variable variance) heteroscedastic (power in the mean)
	IVP Solvers	> Explicit > Implicit > Programming	Runge-Kutta type methods: RKF45; Adams method: ode113 Matlab Methods for stiff systems: Radau5, ode15s Matlab, LSODA, LSODES Mex-files to FORTRAN compiled code are used to accelerate simulation
	NLP Solvers	> Local methods > Multistart of local > Global /hybrid	fmincon, fsqp, ipopt, nomad, dhc, fminsearch, ... SSm, DE, SRES, sequential hybrids: DE+local and SRES+local
Outputs		> Report > Structure > Figures	.m file that keeps input data and optimal experimental design .mat that keeps inputs. and results. structures .fig [and .eps] files with the optimal experimental scheme and the corresponding correlation matrix

Optimal experimental design

Inputs

```
% PATHS RELATED DATA
%.... as in previous slides
% MODEL RELATED DATA
%.... as in previous slides
% EXPERIMENTAL SCHEME RELATED DATA
inputs.exps.n_exp=3;
inputs.exps.exp_type{1}='fixed';
inputs.exps.exp_type{2}='fixed';
inputs.exps.exp_type{3}='od';
% EXPERIMENTS 1 & 2: FIXED
for iexp=1:2
inputs.exps.n_obs{iexp}=2; inputs.exps.obs_names{iexp}=char('Lum','mRNAa');
inputs.exps.obs{iexp}=char('Lum=CL_m','mRNAa=CT_m');
inputs.exps.exp_y0{iexp}=zeros(1,inputs.model.n_st);
end
inputs.exps.u_interp{1}='sustained'; inputs.exps.t_f{1}=120;
inputs.exps.t_con{1}=[0 120]; inputs.exps.u{1}=[1]; inputs.exps.n_s{1}=15;
```

```
inputs.exps.u_interp{2}='pulse-down'; inputs.exps.n_pulses{2}=5;
inputs.exps.u_min{2}=0; inputs.exps.u_max{2}=1;
inputs.exps.t_f{2}=120; inputs.exps.t_con{2}=[0 :12: 120]; inputs.exps.n_s{2}=25;
```

! % INPUTS FOR THE EXPERIMENT TO BE OPTIMALLY DESIGNED

```
inputs.exps.n_obs{3}=2; inputs.exps.obs_names{3}=char('Lum','mRNAa');
inputs.exps.obs{3}=char('Lum=CL_m','mRNAa=CT_m');
inputs.exps.exp_y0{3}=zeros(1,inputs.model.n_st);
inputs.exps.u_type{3}='od';
inputs.exps.u_interp{3}='pulse-up'; inputs.exps.n_pulses{3}=2;
inputs.exps.u_min{3}=0; inputs.exps.u_max{3}=1;
inputs.exps.tf_type{3}='fixed'; inputs.exps.t_f{3}=120;
inputs.exps.ts_type{3}='fixed'; inputs.exps.n_s{3}=10;
```

% PARAMETERS TO BE CONSIDERED FOR OED

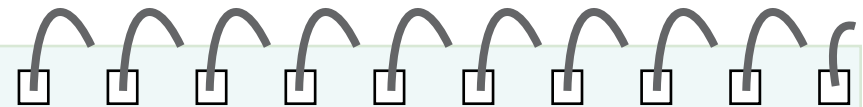
```
inputs.PEsol.id_global_theta=char('n1','n2','m1','m4','m6','m7','k1','k4','p3');
inputs.PEsol.global_theta_guess=[6.1307 7.0943e-001 8.9540 2.3850 2.1019...
1.2904 4.305 2.4561 4.9036e-001];
```

% COST FUNCTION RELATED DATA

```
inputs.exps.noise_type='homo_var';
inputs.OEDsol.OEDcost_type='Dopt';
```

Illustrative example: The circadian clock in *Arabidopsis thaliana*

Design of one pulse-up experiment under homoscedastic noise. The two previous experiments are being considered for the FIM computation. Global solution found for the model unknowns is used. D-optimality criterion is selected.



How to design a new experiment ?

% By default ssm is being used....

```
>> AMIGO_OED('circadian_oed')
```

```
>> AMIGO_OED('circadian_oed','run2','multi_fmincon')
```

Optimal experimental design

Outputs

Illustrative example: The circadian clock in *Arabidopsis thaliana*

circadian-tutorial

- Contours_circadian_run1
- GRank_circadian_run1
- LRank_circadian_run1
- OED_circadian_ssm_run1
- PE_circadian_de_run1
- PE_circadian_de_run2
- PE_circadian_hyb_de_dn2fb_p1
- PE_circadian_hyb_de_fmincon_p1
- PE_circadian_multi_fmincon_run1
- PE_circadian_ssm_run1
- RIdent_circadian_ssm_run1
- SData_circadian_real
- SData_circadian_pseudo
- SModel_circadian_run1
- SObs_circadian_run1

- circadian_oed_input_run1.m
- conv_curve.fig
- corr_mat.fig
- obs_plot_exp1_1.fig
- obs_plot_exp2_1.fig
- obs_plot_exp3_1.fig
- report_circadian_real.m
- strreport_circadian_real.mat

```
>> load strreport_circadian_run1.mat

>> results

results =

    pathd: [1x1 struct]
    plotd: [1x1 struct]
    nlpsol: [1x1 struct]
    oed: [1x1 struct]
    sim: [1x1 struct]

>> >> results.oed

    n_exp: 3
    n_obs: {[2] [2] [2]}
    obs: {[2x10 char] [2x10 char] [2x10 char]}
    n_s: {[15] [25] [10]}
    t_s: {[1x15 double] [1x25 double] [0 13.3333 26.6667 40 53.3333 66.6667 80 93.3333 106.6667 120]}
    u: {[1] [1 0 1 0 1 0 1 0 1 0] [0 1 0 1 0]}
    t_con: {[0 120] [0 12 24 36 48 60 72 84 96 108 120] [0 12.6385 39.5065 65.9786 90.5559 120]}
    exp_y0: {[0 0 0 0 0 0 0] [0 0 0 0 0 0 0] [0 0 0 0 0 0 0]}
    w_sampling: {1x3 cell}
    sens_t: {[15x2x7 double] [25x2x7 double] [10x2x7 double]}
    r_sens_t: {[15x2x7 double] [25x2x7 double] [25x2x7 double]}
    ms: {[15x2 double] [25x2 double] [10x2 double]}
    g_FIM: [7x7 double]
    g_corr_mat: [8x8 double]
    conf_intervals: [0.3847 0.5724 0.1744 0.2460 0.2166 0.0095 0.1055]
```

Overall experimental scheme

Sensitivities, FIM, correlation matrix and confidence intervals

Optimal experimental design

Outputs

```
.m circadian_oed_input_run1.m  
.fig conv_curve.fig  
.fig corr_mat.fig  
.fig obs_plot_exp1_1.fig  
.fig obs_plot_exp2_1.fig  
.fig obs_plot_exp3_1.fig  
.m report_circadian_real.m  
.mat strreport_circadian_real.mat
```

Illustrative example: The circadian clock in *Arabidopsis thaliana*

Reported Crammer Rao confidence intervals before and after OED

```
>>> Estimated global parameters:
```

```
n1 : 6.1306e+000 +- 1.6186e+000;  
n2 : 7.0943e-001 +- 4.0134e-002;  
m1 : 8.9541e+000 +- 1.4441e+000;  
m4 : 2.3850e+000 +- 1.9231e-001;  
m6 : 2.1020e+000 +- 4.4199e-001;  
m7 : 1.2903e+000 +- 8.1317e-001;  
k1 : 4.3052e+000 +- 1.0059e+000;  
k4 : 2.4560e+000 +- 3.0707e-001;  
p3 : 4.9028e-001 +- 3.5386e-001;
```

```
>>> OED corresponding Cramer Rao expected  
uncertainty for the unknowns:
```

```
>>> Global parameters:
```

```
n1 : 6.1307e+000 +- 7.5405e-001;  
m1 : 8.9540e+000 +- 1.1218e+000;  
m6 : 2.1019e+000 +- 3.4184e-001;  
m7 : 1.2904e+000 +- 4.8212e-001;  
k1 : 4.3050e+000 +- 4.2463e-001;  
k4 : 2.4561e+000 +- 1.8681e-002;  
p3 : 4.9036e-001 +- 2.0675e-001;
```

