Non-linear mixed effects modelling using the SAS system - An overview

17 June 2004

Al Maloney

What this talk hopes to show...

Fitting non-linear mixed effect models in SAS using the NLMIXED procedure.

- Background
- The syntax - defining the model
- The options - defining the criteria for fitting the model
- Strengths and Limitations
- Summary
- (If time, a very brief example using NLMIXED - adaptive design)

This outline is based on the SAS online documentation !

A brief note on the nomenclature used

General term	NM term
Fixed effects	Thetas
Random effects	Etas
Var-Cov of random effects	Omega

SAS seem to have incorporated the current 'best' knowledge on NL mixed models methods

Built on work from a number of 'different' fields including:

▪ Beal, Sheiner...	Mixed models in PK / PK-PD
▪ Goldstein...	Hierarchical mixed models
▪ Longford, Diggle...	Generalised linear mixed models
▪ Lindstrom, Bates, Pinheiro...	General applied mixed models
▪ Davidian, Giltinan...	Heteroscedastistic/NP mixed models

....

+ SAS's own knowledge as well !

This has been 'packaged' in a new SAS procedure - PROC NLMIXED

PROC NLMIXED has a number of program statements which can be used

The following statements can be used with the NLMIXED procedure:

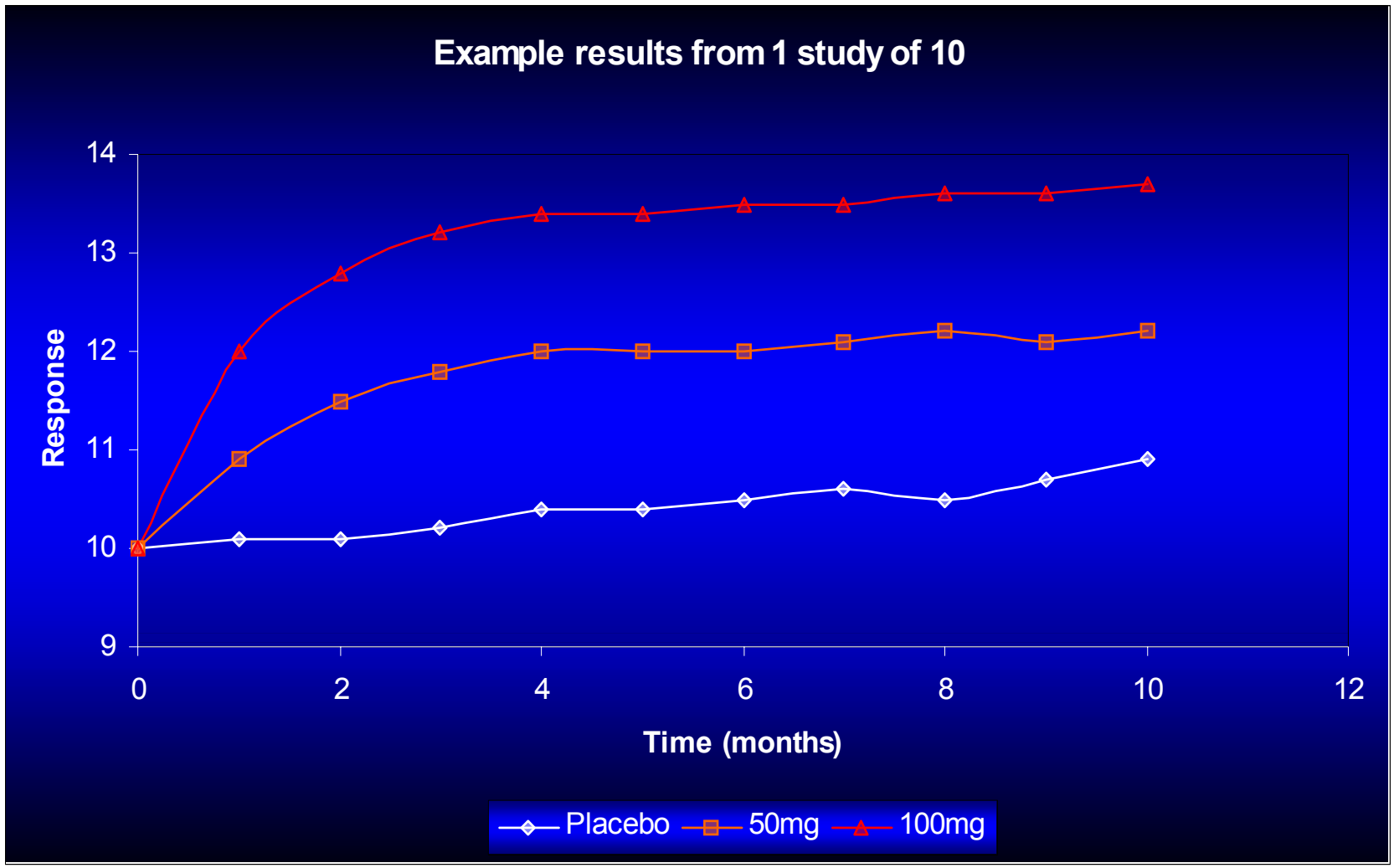
PROC NLMIXED options	**procedure call and options**
ARRAY	**array specification**
PARMS	**parameters and starting values**
BOUNDS	**boundary constraints**
BY	**variables**
CONTRAST	**'label' expression**
ID	**expressions**
MODEL	**model specification**
RANDOM	**random effects specification**
PREDICT	**expression**
ESTIMATE	**'label' expression**
ODS	**output control**
REPLICATE	**replicate variable**
Program statements	**usual SAS programming statements**
Run;	

This talk will focus on the main statements

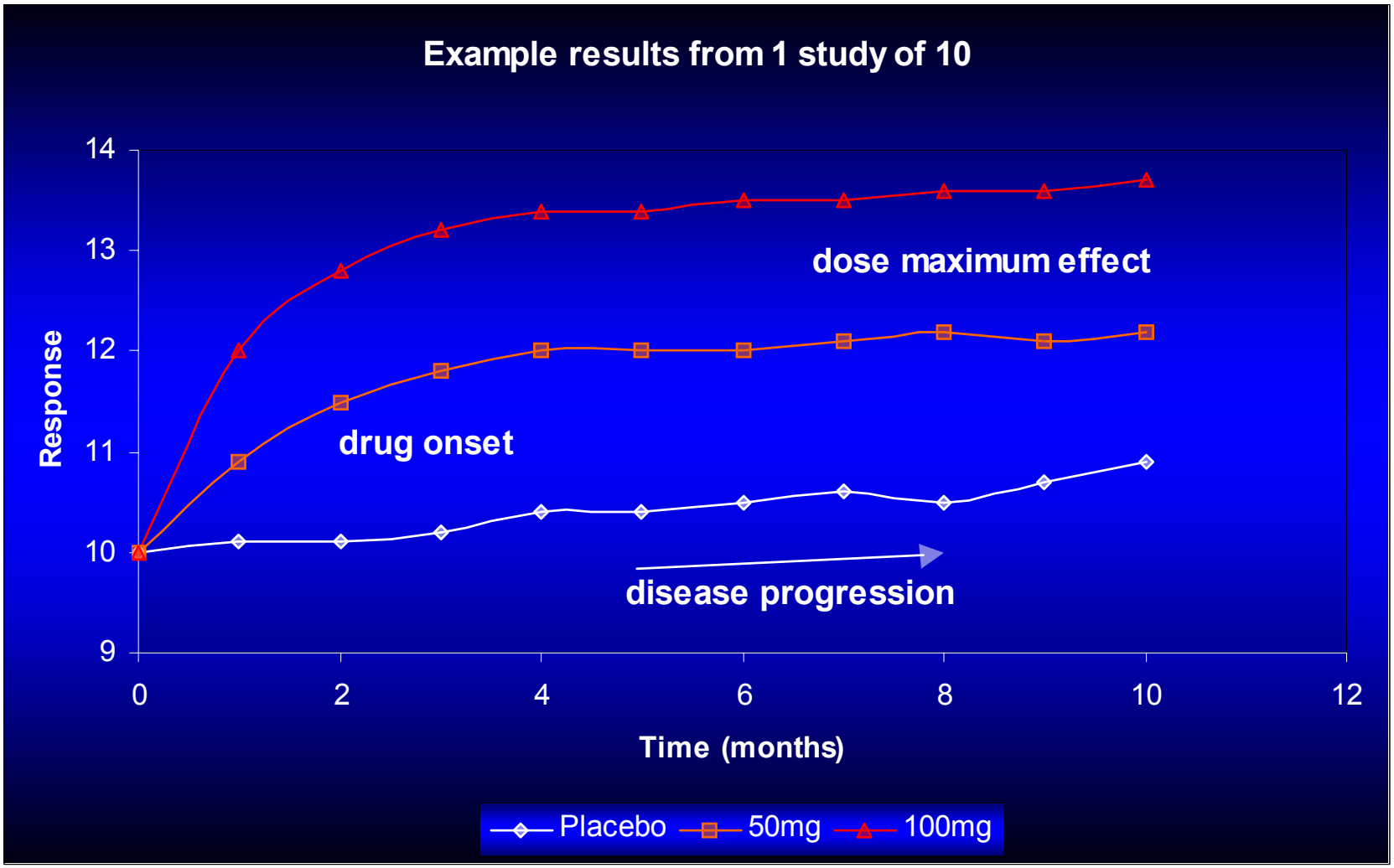
The most important statements are:

PROC NLMIXED options	**procedure call and options**
ARRAY	**array specification**
PARMS	**parameters and starting values**
BOUNDS	**boundary constraints**
BY	**variables**
CONTRAST	**'label' expression**
ID	**expressions**
MODEL	**model specification**
RANDOM	**random effects specification**
PREDICT	**expression**
ESTIMATE	**'label' expression**
ODS	**output control**
REPLICATE	**replicate variable**
Program statements	**usual SAS programming statements**
Run;	

To help interpret the programming statements, consider some artificial data



To help interpret the programming statements, consider some artificial data



The dataset to be used in the NLMIXED procedure has the relevant data

Data PAGE;

...

Input	Study	Drug	Dose	Time	Observed
...					
	1	A	50	0	10.0
	1	A	50	1	10.9
....					
	2	B	0	0	10.0

10 Studies (random effect level)

3 Drugs ("A" "B" and "C")

For this artificial drug disease model, the SAS code might look like:

options

```
proc nlmixed data=page cov corr method=firo tech=quanew gtol=1E-10 ftol=1E-10;
parms  LED50_A=1 LED50_B=2 LED50_C=3 EMAX=0.5 SLOPE = 1 to 5 by 1
       RATE = 0.5 b1var =1 b2var 1 s2= 1;
bounds RATE>0;

    Disease    = 10 + 0.1*(SLOPE+b11)*time;
    if drug    = "A" then LED50=LED50_A;
    if drug    = "B" then LED50=LED50_B;
    if drug    = "C" then LED50=LED50_C;
    drug_eff   = (EMAX+b22) * dose / (exp(LED50)+dose);
    drug_delay = 1-exp(-RATE*time);
    pred       = disease*(1+drug_eff*drug_delay);

model observed ~ normal(pred, s2);
random b11 b22 ~ normal([0,0],[b11var, 0, b22var]) subject=study out =checkran1;
ods output ParameterEstimates = est1; ods output CovMatParmEst=cov1;
run;
```

disease

drug

delay

response

For this artificial drug disease model, the SAS code might look like:

options

```
proc nlmixed data=test cov corr method=firo tech=quanew gtol=1E-10 ftol=1E-10;
parms  LED50_A=1 LED50_B=2 LED50_C=3 EMAX=0.5 SLOPE = 1 to 5 by 1
       RATE = 0.5 b1var =1 b2var 1 s2= 1;
bounds RATE>0;
```

```
    Disease    = 10 + 0.1*(SLOPE+b11)*time;
    if drug    = "A" then LED50=LED50_A;
    if drug    = "B" then LED50=LED50_B;
    if drug    = "C" then LED50=LED50_C;
    drug_eff   = (EMAX+b22) * dose / (exp(LED50)+dose);
    drug_delay = 1-exp(-RATE*time);
    pred       = disease*(1+drug_eff*drug_delay);
```

disease

drug

delay

response

```
model observed ~ normal(pred, s2);
random b11 b22 ~ normal([0,0],[b11var, 0, b22var]) subject=study out =checkran1;
ods output ParameterEstimates = est1; ods output CovMatParmEst=cov1;
run;
```

The "PARMS" statement defines starting points or initial grid search for each parameter

```
proc nlmixed data=test cov corr method=firo tech=quanew gtol=1E-10 ftol=1E-10;
parms  LED50_A=1 LED50_B=2 LED50_C=3 EMAX=0.5 SLOPE = 1 to 5 by 1
       RATE = 0.5 b1var =1 b2var 1 s2= 1;
bounds RATE>0;

    Disease    = 10 + 0.1*(SLOPE+b11)*time;
    if drug    = "A" then LED50=LED50_A;
    if drug    = "B" then LED50=LED50_B;
    if drug    = "C" then LED50=LED50_C;
    drug_eff   = (EMAX+b22) * dose / (exp(LED50)+dose);
    drug_delay = 1-exp(-RATE*time);
    pred       = disease*(1+drug_eff*drug_delay);

model observed ~ normal(pred, s2);
random b11 b22 ~ normal([0,0],[b11var, 0, b22var]) subject=study out =checkran1;
ods output ParameterEstimates = est1; ods output CovMatParmEst=cov1;
run;
```

disease

drug

delay

response

The "PARMS" statement defines starting points or initial grid search for each parameter

The "PARMS" statement gives initial estimates for the model parameters. You can also give a range of potential values. SAS will perform the grid search, and start the optimisation at the best combination.

Example:

```
parms  LED50_A=1 LED50_B=2 LED50_C=3 EMAX=0.5
slope = 1 to 5 by 1 b1var=1 b2var=1 s2= 1 RATE = 0.5;
```

- Note the use of "TO" and "BY".
- Can use this to simply plot the likelihood function (no fitting).
- Can read in previous model fit parameters.

The "BOUNDS" and "BY" statements do exactly what you would expect.

```
proc nlmixed data=test cov corr method=firo tech=quanew gtol=1E-10 ftol=1E-10;
parms  LED50_A=1 LED50_B=2 LED50_C=3 EMAX=0.5 SLOPE = 1 to 5 by 1
       RATE = 0.5 b1var =1 b2var 1 s2= 1;
bounds RATE>0;

    Disease    = 10 + 0.1*(SLOPE+b11)*time;
    if drug    = "A" then LED50=LED50_A;
    if drug    = "B" then LED50=LED50_B;
    if drug    = "C" then LED50=LED50_C;
    drug_eff   = (EMAX+b22) * dose / (exp(LED50)+dose);
    drug_delay = 1-exp(-RATE*time);
    pred       = disease*(1+drug_eff*drug_delay);

model observed ~ normal(pred, s2);
random b11 b22 ~ normal([0,0],[b11var, 0, b22var]) subject=study out =checkran1;
ods output ParameterEstimates = est1; ods output CovMatParmEst=cov1;
run;
```

disease

drug

delay

response

The "BOUNDS" and "BY" statements do exactly what you would expect.

The "BOUNDS" statement

- Limits on any parameters (although should generally be avoided with model reparameterisation and/or reduction).

Example:

```
bounds RATE>0;
```

The "BY" statement

- Useful for repeated model fitting (e.g. bootstrap samples)

Example :

```
by boot_sample;
```

The "MODEL" statement allows a wide variety of models, including defining your own log likelihood

```
proc nlmixed data=test cov corr method=firo tech=quanew gtol=1E-10 ftol=1E-10;
parms  LED50_A=1 LED50_B=2 LED50_C=3 EMAX=0.5 SLOPE = 1 to 5 by 1
       RATE = 0.5 b1var =1 b2var 1 s2= 1;
bounds RATE>0;

    Disease    = 10 + 0.1*(SLOPE+b11)*time;
    if drug    = "A" then LED50=LED50_A;
    if drug    = "B" then LED50=LED50_B;
    if drug    = "C" then LED50=LED50_C;
    drug_eff   = (EMAX+b22) * dose / (exp(LED50)+dose);
    drug_delay = 1-exp(-RATE*time);
    pred       = disease*(1+drug_eff*drug_delay);

model observed ~ normal(pred, s2);
random b11 b22 ~ normal([0,0],[b11var, 0, b22var]) subject=study out =checkran1;
ods output ParameterEstimates = est1; ods output CovMatParmEst=cov1;
run;
```

disease

drug

delay

response

The "MODEL" statement allows a wide variety of models, including defining your own log likelihood

The "MODEL" statement defines the type of likelihood function.

Valid distributions are as follows.

- *normal(m,v)* specifies a normal distribution with mean m and variance v.
- *binary(p)* specifies a binary (Bernouilli) distribution with probability p.
- *binomial(n,p)* specifies a binomial distribution with count n and probability p.
- *poisson(m)* specifies a Poisson distribution with mean m.
- *general(ll)* specifies a general log likelihood function that you define.

Examples

```
model observed ~ normal(pred, s2);
```

or equivalently

```
ll=-.5*log(2*3.14159265358979*s2)-(.5/s2)*(y-pred)**2;

model y~general(ll);
```

The "RANDOM" statement defined the variance covariance matrix of random effects

```
proc nlmixed data=test cov corr method=firo tech=quanew gtol=1E-10 ftol=1E-10;
parms  LED50_A=1 LED50_B=2 LED50_C=3 EMAX=0.5 SLOPE = 1 to 5 by 1
       RATE = 0.5 b1var =1 b2var 1 s2= 1;
bounds RATE>0;

    Disease    = 10 + 0.1*(SLOPE+b11)*time;
    if drug    = "A" then LED50=LED50_A;
    if drug    = "B" then LED50=LED50_B;
    if drug    = "C" then LED50=LED50_C;
    drug_eff   = (EMAX+b22) * dose / (exp(LED50)+dose);
    drug_delay = 1-exp(-RATE*time);
    pred       = disease*(1+drug_eff*drug_delay);

model observed ~ normal(pred, s2);
random b11 b22 ~ normal([0,0],[b11var, 0, b22var]) subject=study out =checkran1;
ods output ParameterEstimates = est1; ods output CovMatParmEst=cov1;
run;
```

disease

drug

delay

response

The "RANDOM" statement defined the variance covariance matrix of random effects

Define (lower) diagonal and off-diagonal random elements that need to be estimated.

e.g. Simple two random effects, no correlation

$$\begin{pmatrix} \Omega^{11} & 0 \\ 0 & \Omega^{22} \end{pmatrix}$$

```
random b11 b22 ~ normal([0,0],[b11var, 0, b22var]) subject=study out =checkran1;
```

e.g. $$\begin{pmatrix} \Omega^{11} & \Omega^{12} & \Omega^{13} & 0 \\ \Omega^{12} & \Omega^{22} & \Omega^{23} & \Omega^{24} \\ \Omega^{13} & \Omega^{23} & \Omega^{33} & 0 \\ 0 & \Omega^{24} & 0 & \Omega^{44} \end{pmatrix}$$

```
random b1 b2 b3 b4 ~ normal([0,0,0,0],[Ω11,Ω12,Ω22,Ω13,Ω23,Ω33,0,Ω24,0,Ω44])
subject=study;
```

The "RANDOM" statement defined the variance covariance matrix of random effects

Define (lower) diagonal and off-diagonal random elements that need to be estimated.

e.g. Simple two random effects, no correlation

$$\begin{pmatrix} \Omega^{11} & 0 \\ 0 & \Omega^{22} \end{pmatrix}$$

```
random b11 b22 ~ normal([0,0],[b11var, 0, b22var]) subject=study out =checkran1;
```

e.g.

$$\begin{pmatrix} \Omega^{11} & \Omega^{12} & \Omega^{13} & 0 \\ \Omega^{12} & \Omega^{22} & \Omega^{23} & \Omega^{24} \\ \Omega^{13} & \Omega^{23} & \Omega^{33} & 0 \\ 0 & \Omega^{24} & 0 & \Omega^{44} \end{pmatrix}$$

```
random b1 b2 b3 b4 ~ normal([0,0,0,0],[Ω11,Ω12,Ω22,Ω13,Ω23,Ω33,0,Ω24,0,Ω44])
subject=study;
```

The Output Deliver System (ODS) in SAS allows any output to be available later

```
proc nlmixed data=test cov corr method=firo tech=quanew gtol=1E-10 ftol=1E-10;
parms  LED50_A=1 LED50_B=2 LED50_C=3 EMAX=0.5 SLOPE = 1 to 5 by 1
       RATE = 0.5 b1var =1 b2var 1 s2= 1;
bounds RATE>0;

    Disease    = 10 + 0.1*(SLOPE+b11)*time;
    if drug    = "A" then LED50=LED50_A;
    if drug    = "B" then LED50=LED50_B;
    if drug    = "C" then LED50=LED50_C;
    drug_eff   = (EMAX+b22) * dose / (exp(LED50)+dose);
    drug_delay = 1-exp(-RATE*time);
    pred       = disease*(1+drug_eff*drug_delay);

model observed ~ normal(pred, s2);
random b11 b22 ~ normal([0,0],[b11var, 0, b22var]) subject=study out =checkran1;
ods output ParameterEstimates = est1; ods output CovMatParmEst=cov1;
run;
```

disease

drug

delay

response

The Output Deliver System (ODS) in SAS allows any output to be available later

Any output that is can be written to the results file can also be saved to a dataset for subsequent manipulation. This includes:

- Parameter estimates
- Fit statistics
- Model specification
- Covariance matrix
- Correlation matrix
- Convergence status
- etc.

e.g.

```
ods output ParameterEstimates = est1; ods output CovMatParmEst=cov1;
```

Some additional nice features

- Most SAS programming statements (e.g. "If..Then") are supported.

```
if drug = "A" then LED50=LED50_A;
if drug = "B" then LED50=LED50_B;
if drug = "C" then LED50=LED50_C;
```

- Using previous model parameters (reading in the last fit)

```
ods output ParameterEstimates=pe;      * (create dataset with estimated parameters);
```

...then on the next call...

```
parms / data=pe;                       * (read in dataset with previous estimates);
```

Options in the model fitting

Where SAS shows it's strength, and the complexities of fitting these models !

Some of the main options are:

Basic Options

DATA=input data set

METHOD=integration method

Displayed Output Specifications

START=gradient at starting values

HESS=Hessian matrix
ITDETAILS =iteration details
CORR=correlation matrix
COV=covariance matrix
ECORR=corr matrix of additional estimates
ECOV=cov matrix of additional estimates
EDER=derivatives of additional estimates
ALPHA==alpha for confidence limits
DF=degrees of freedom for p values and
confidence limits

Derivatives Specifications

FD[=]finite-difference derivatives
FDHESSIAN[=]finite-difference second derivatives
DIAHES=use only diagonal of Hessian

Optimisation Specifications

TECHNIQUE=minimization technique
UPDATE=update technique
LINESEARCH=line-search method
LSPRECISION=line-search precision
HESCAL=type of Hessian scaling
INHESSIAN=start for approximated Hessian
RESTART=iteration number for update restart
OPTCHECK[=]check optimality in neighbourhood

Termination Criteria Specifications

MAXFUNC=maximum number of function calls
MAXITER=maximum number of iterations
MINITER=minimum number of iterations
MAXTIME=upper limit seconds of CPU time
ABSCONV=absolute function convergence criterion
ABSFCONV=absolute function convergence criterion
ABSGCONV=absolute gradient convergence criterion
ABSXCONV=absolute parameter convergence criterion
FCONV=relative function convergence criterion
FCONV2=relative function convergence criterion
GCONV=relative gradient convergence criterion
XCONV=relative parameter convergence criterion
FDIGITS=number accurate digits in objective function
FSIZE=used in FCONV, GCONV criterion
XSIZE=used in XCONV criterion

Additional refinement and debugging of the model fitting can be achieved with other options

Quadrature Options
NOAD=no adaptive centring
NOADSCALE=no adaptive scaling
OUTQ=output data set
QFAC=search factor
QMAX=maximum points
QPOINTS=number of points
QSCALEFAC=scale factor
QTOL=tolerance

Debugging Output
LIST=model program, variables
LISTCODE=compiled model program
LISTDEP=model dependency listing
LISTDER=model derivative
XREF=model cross reference
FLOW=model execution messages
TRACE=detailed model execution messages

Singularity Tolerances
SINGCHOL=tolerance for Cholesky roots
SINGHESS=tolerance for Hessian
SINGSWEEP=tolerance for sweep
SINGVAR=tolerance for variances

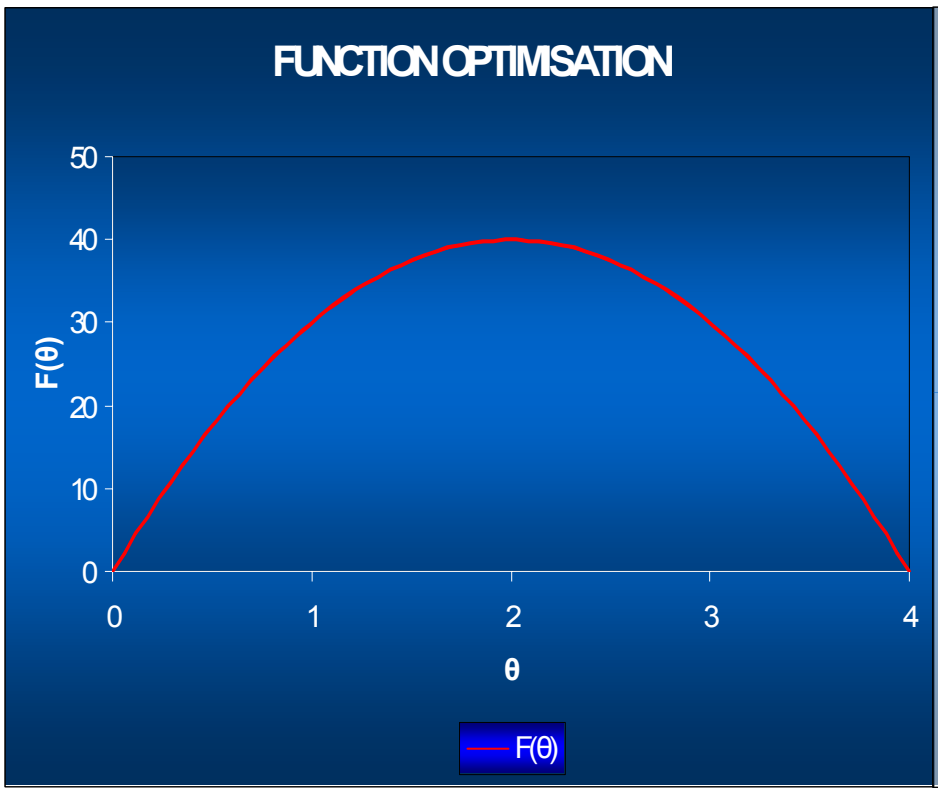
Empirical Bayes Options
EBSTEPS=number of Newton steps
EBSUBSTEPS=number of sub steps
EBSSFRAC=step-shortening fraction
EBSSTOL=step-shortening tolerance
EBTOL=convergence tolerance
EBOPT= comprehensive optimisation
EBZSTART=zero starting values

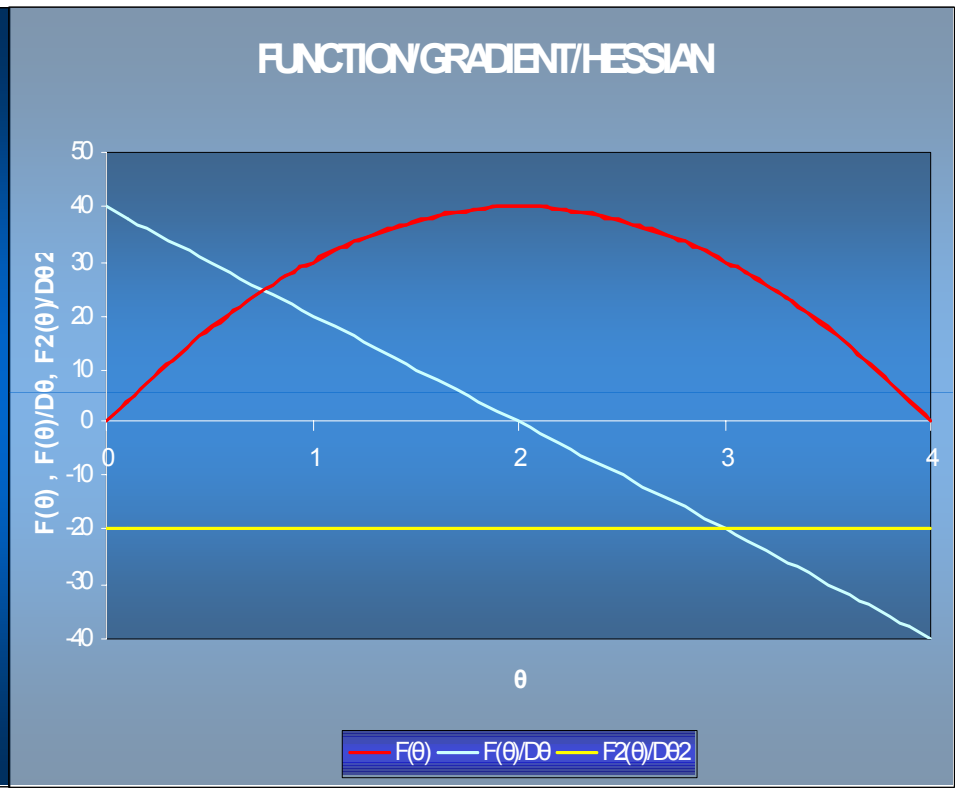
Step Length Specifications
DAMPSTEP[=]damped steps in line search
MAXSTEP=maximum trust-region radius
INSTEP=initial trust-region radius

Covariance Matrix Tolerances
ASINGULAR=absolute singularity for inertia
MSINGULAR=relative M singularity for inertia
VSINGULAR=relative V singularity for inertia
G4=threshold for Moore-Penrose inverse
COVSING=tolerance for singular COV matrix
CFACTOR=multiplication factor for COV matrix

Constraint Specifications
LCEPSILON=range for active constraints
LCDEACT=LM tolerance for deactivating
LCSINGULAR=tolerance for dependent constraints

A brief reminder of function optimisation and gradient/hessian terms





Optimisation Algorithms

7 function optimisation methods

		Needs	
		Gradient	Hessian
quasi-Newton (DBFGS, DDFP, BFGS, DFP)	QUANEW *	Yes	No
trust region	TRUREG	Yes	Yes
Newton-Raphson with line search	NEWRAP	Yes	Yes
Newton-Raphson with ridging	NRRIDG	Yes	Yes
double-dogleg (DBFGS, DDFP)	DBLDOG	Yes	No
conjugate gradient (PB, FR, PR, CD)	CONGRA	Yes	No
Nelder-Mead method	NMSIMP	No	No

*default

Finite difference approximations of derivatives - forward or central?

SAS uses numerical approximations for derivatives.

Gradient - first order derivatives - rate of change of function

Hessian - second order derivatives - rate of change of rate of change

Consider the gradient:

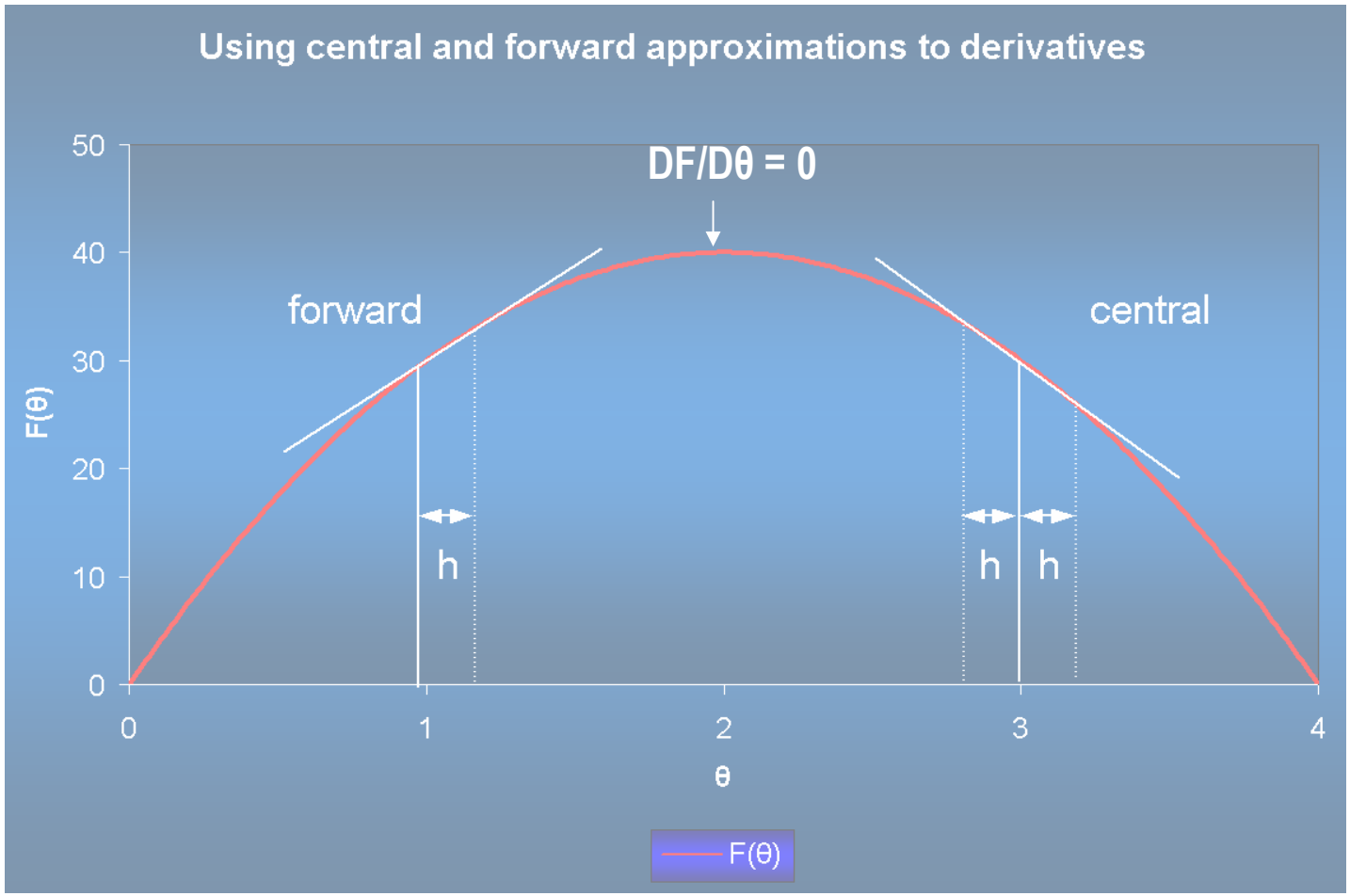
Forward

$$g_i = \frac{\partial f}{\partial \theta_i} \approx \frac{f(\theta + h_i e_i) - f(\theta)}{h_i}$$

Central

$$g_i = \frac{\partial f}{\partial \theta_i} \approx \frac{f(\theta + h_i e_i) - f(\theta - h_i e_i)}{2h_i}$$

A quick reminder of forward and central numerical derivatives calculations



Integral Approximations

Likelihood $$m(\theta) = \prod_{i=1}^{s} \int p(y_i|X_i, \phi, u_i) q(u_i|\xi) du_i$$

- First Order (as per NM)

  Taylors series expansion around u_i=0

  Only normal data

- Adaptive gaussian quadrature (default)

  Centres integral at u_i, the empirical bayes estimate

  Can choose number of quadrature points (1 = Laplacian approx.)

Termination Criteria - Convergence limits and diagnostics

Convergence is something you decide...not the computer package!

QUANEW algorithm will converge if any of the following are satisfied:

1. ABSGCONV < 10^{-5}
2. FCONV < 10^{-16} *
3. or GCONV < 10^{-8}

* based on machine precision (= 10^{-16} on my computer)

1) = Absolute gradient criteria $\max_j |g_j(\theta^{(k)})| \leq r$

2) = Relative function criteria $\frac{|f(\theta^{(k)}) - f(\theta^{(k-1)})|}{\max(|f(\theta^{(k-1)})|, \mathrm{FSIZE})} \leq r$

3) = Relative gradient convergence $\frac{g(\theta^{(k)})^T [H^{(k)}]^{-1} g(\theta^{(k)})}{\max(|f(\theta^{(k)})|, \mathrm{FSIZE})} \leq r$

Limitations!

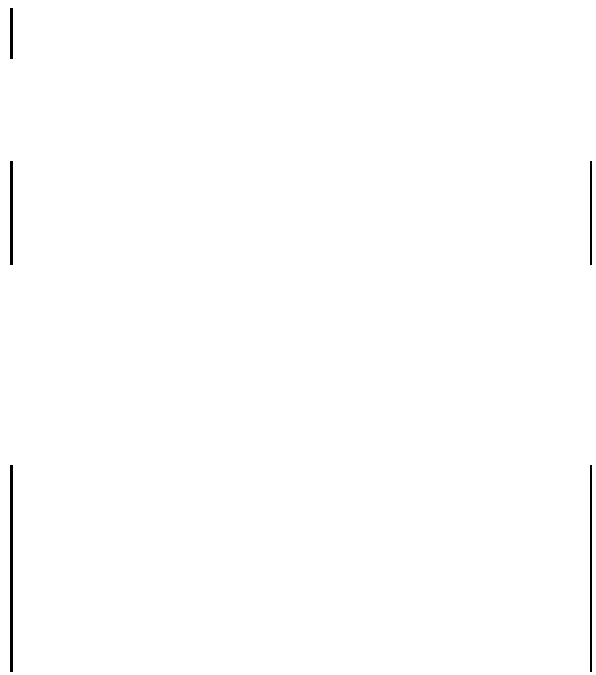
- Only one level of random effect allowed (although can mimic a second level).
- No link with differential equation solvers.
- Complex dosing histories are not accommodated.

Summary

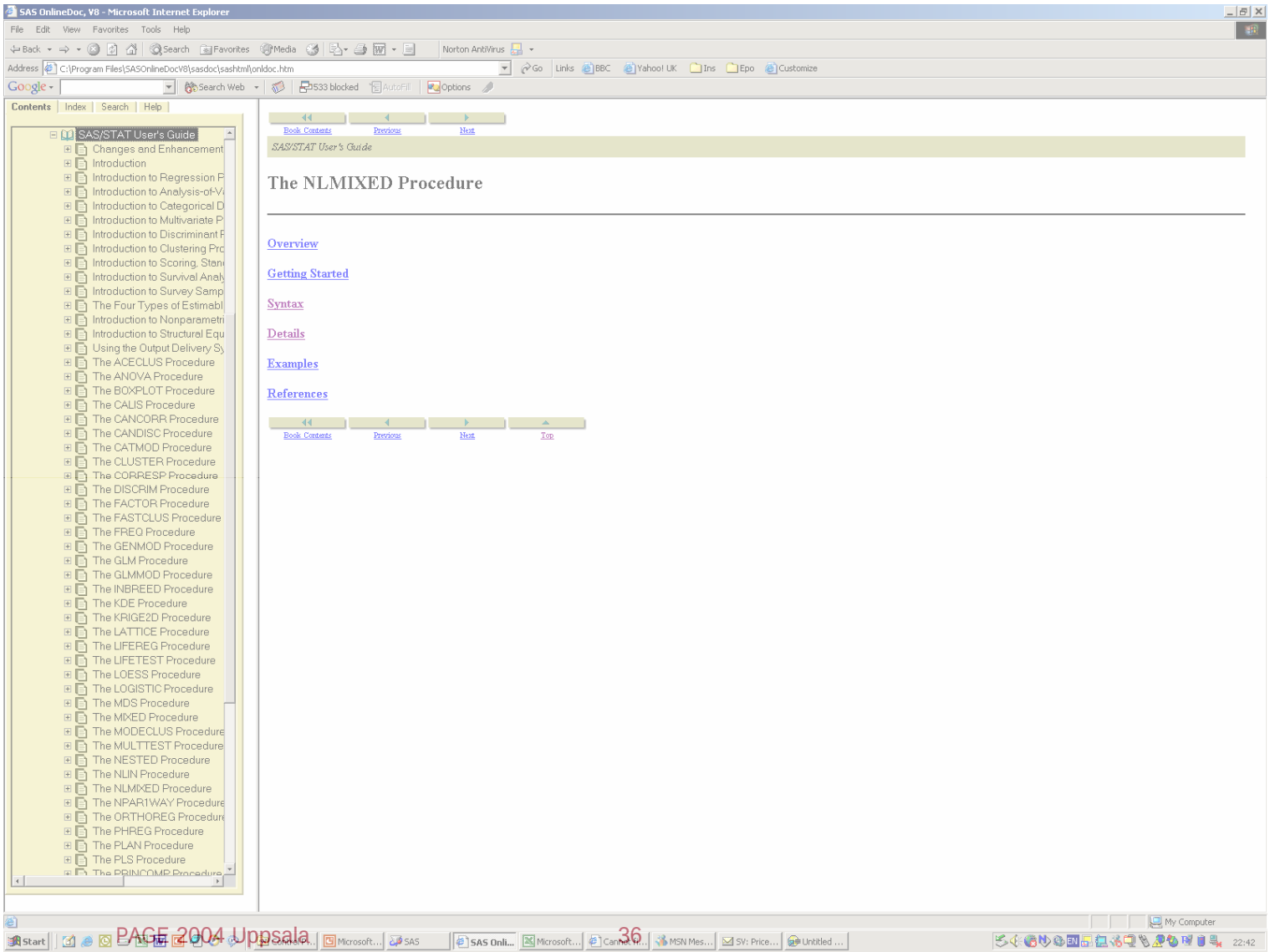
PROC NLMIXED

- Well developed.
- Well documented.
- Easy to use.
- Does what is says it can do, very well.
- Easy access to results.
- Three key limitations limit widespread application within PK/PD

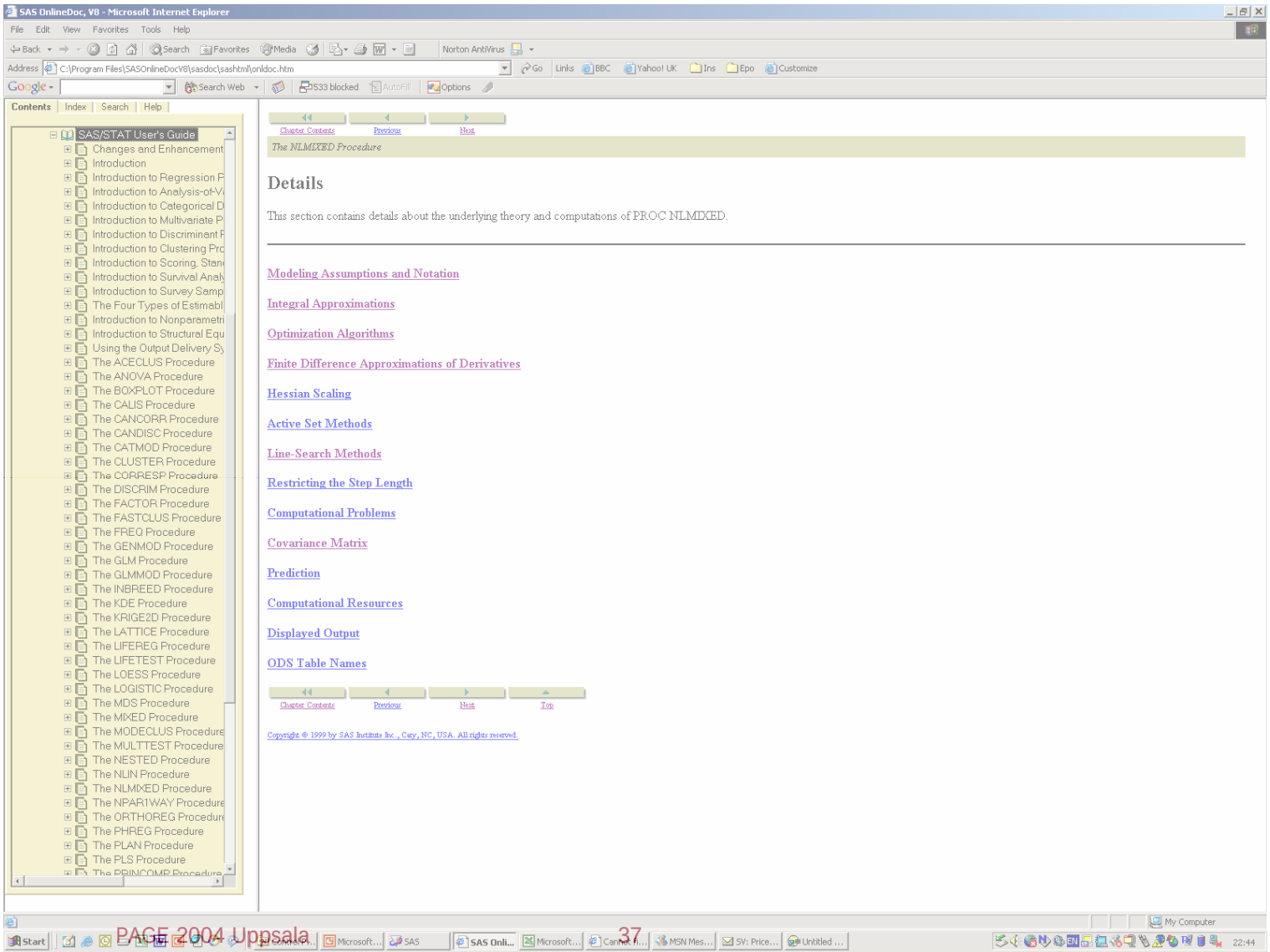
Back up



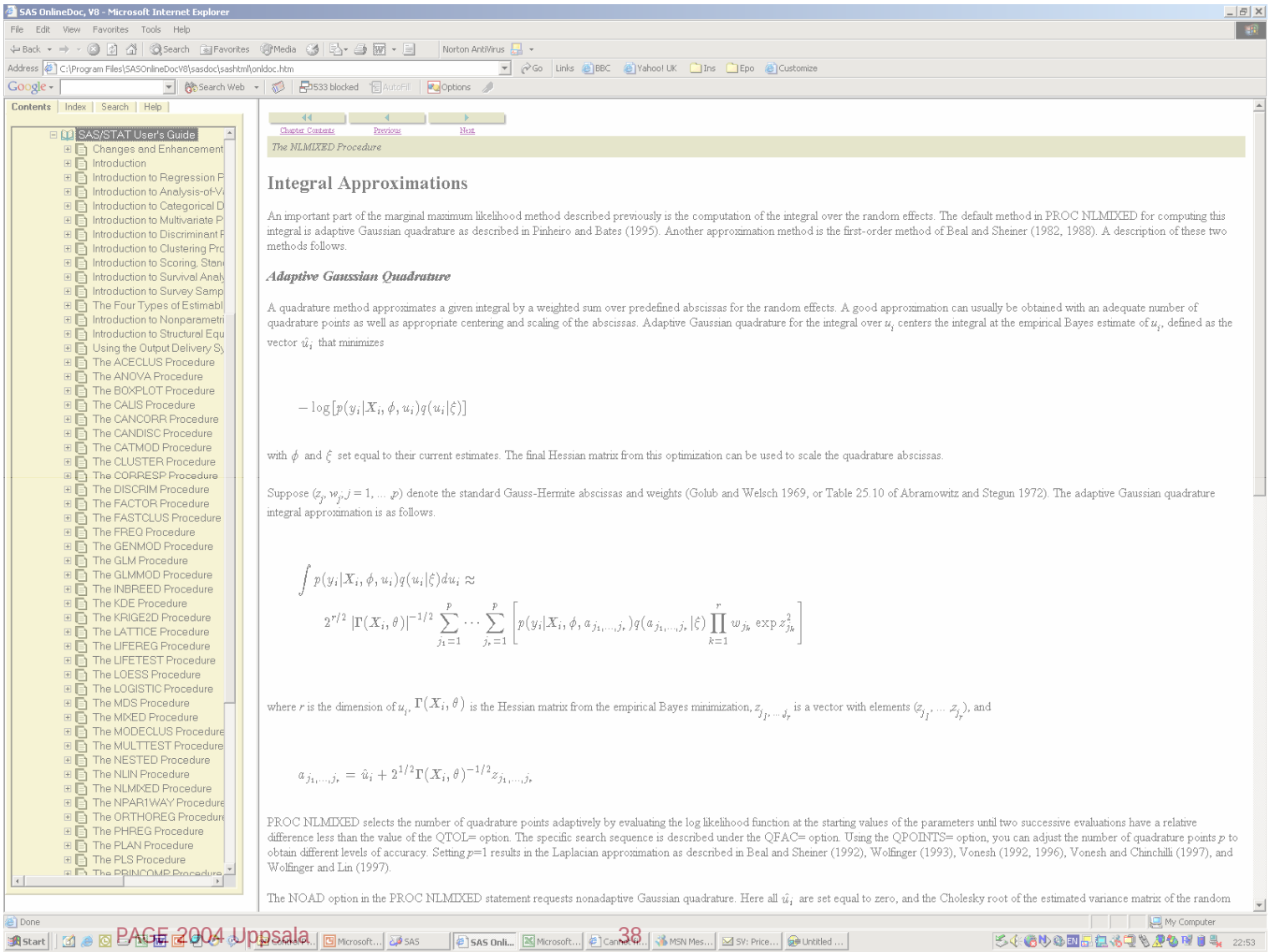
SAS has well documented help files



SAS has well documented help files



SAS has well documented help files



The Output Deliver System (ODS) in SAS allows any output to be available later

ODS Table Name	Description
AdditionalEstimates	Results from ESTIMATE statement
ConvergenceStatus	Convergence status
CorrMatAddEst	Correlation matrix of additional estimates
CorrMatParmEst	Correlation matrix of parameter estimates
CovMatAddEst	Covariance matrix of additional estimates
CovMatParmEst	Covariance matrix of parameter estimates
DerAddEst	Derivatives of additional estimates
Dimensions	Dimensions of the problem
FitStatistics	Fit statistics
Hessian	Second derivative matrix
IterHistory	Iteration history
Parameters	Parameters
ParameterEstimates	Parameter estimates
Specifications	Model specifications
StartingHessian	Starting hessian matrix
StartingValues	Starting values and gradient

e.g.

```
ods output ParameterEstimates = est1; ods output CovMatParmEst=cov1;
```

Finite difference approximations of derivatives - forward or central?

SAS uses numerical approximations for derivatives.

Gradient - first order derivatives

Hessian - second order derivatives

Consider the hessian:

Forward $$\frac{\partial^2 f}{\partial\theta_i\partial\theta_j} \approx \frac{f(\theta + h_i e_i + h_j e_j) - f(\theta + h_i e_i) - f(\theta + h_j e_j) + f(\theta)}{h_i h_j}$$

Central $$\frac{\partial^2 f}{\partial\theta_i^2} \approx \frac{-f(\theta + 2h_i e_i) + 16 f(\theta + h_i e_i) - 30 f(\theta) + 16 f(\theta - h_i e_i) - f(\theta - 2h_i e_i)}{12 h_i^2}$$

The "PREDICT" and "ESTIMATE" statements provide additional requested results

Predict - can obtain predicted results.

predict pred out = pred1;

Estimate - can request additional contrasts

```
pred        = disease (1+drug_eff*drug_delay);
```

placebo = B0 + B1*(baseline);

dose50 = B0 + B1*(baseline) + emax * 50 / (exp(lec50)+50) ;

estimate 'Dose of 50mg v placebo' dose50-plac;

An example

Adaptive design

- Using NLMIXED to determine the 'best' dose to allocate the next patient to, given a tentative model.
- M&S has predicted target dose of Drug X to be between 1 and 200mg.
- 10 potential doses = 0 (placebo), 1, 2, 4, 8, 15, 30, 50,100, 200mg.
- Longitudinal model in place, developed on competitors in similar class.
- Initially, expect ED50 to be different for drug X, but similar Emax.

  Target - require 95% CI for log(ED50) to be within a 4 fold range.

Using NLMIXED, you can determine which is the best dose to randomise the new patient to

Step 1 : Fit model, with data (pseudo + real), and 1 missing records for each potential dose level.

Step 2 : Refit model (10 times) putting in predicted value for each 'virtual' patient at each of the 10 doses.

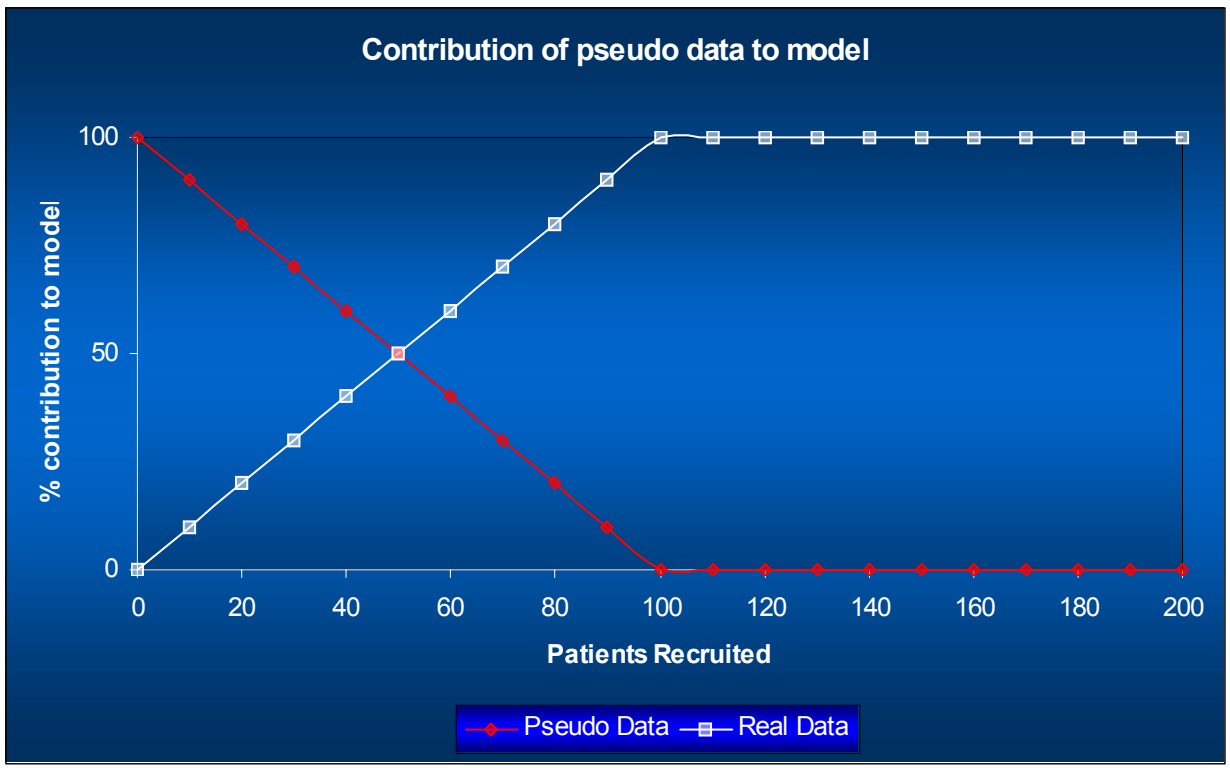
Step 3 : Estimate statistic of choice for each step above.

Step 4: Determine which 'virtual' patient (dose level) provides the most information to your statistic/statistics of interest.

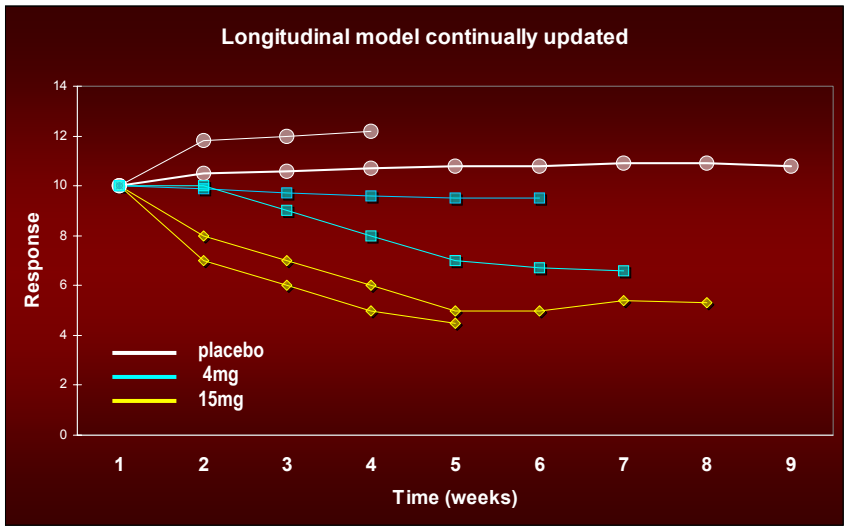
Step 5: Randomise to best dose, and repeat for next patient.

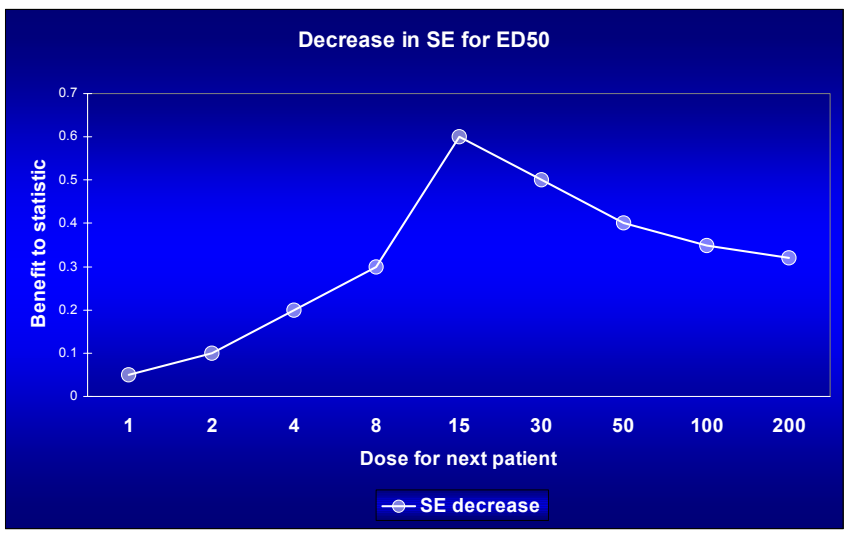
To give the model some initial stability, you create pseudo data.

Akin to priors in a bayesian sense. ED50 will depend on pseudo data at start of recruitment, but will not contribute at end.



At each iteration, the data is updated, and the benefit to the test statistic is determined





As study progresses, we can fit new data into longitudinal model

10 NLMIXED fits, with a new dose of 15mg for the next patient likely to providing the most information

Good coding methods - give the algorithms the best possible chance of being successful

- Rescaling
- Centering
- Reparameterisation
- Eigenvalues