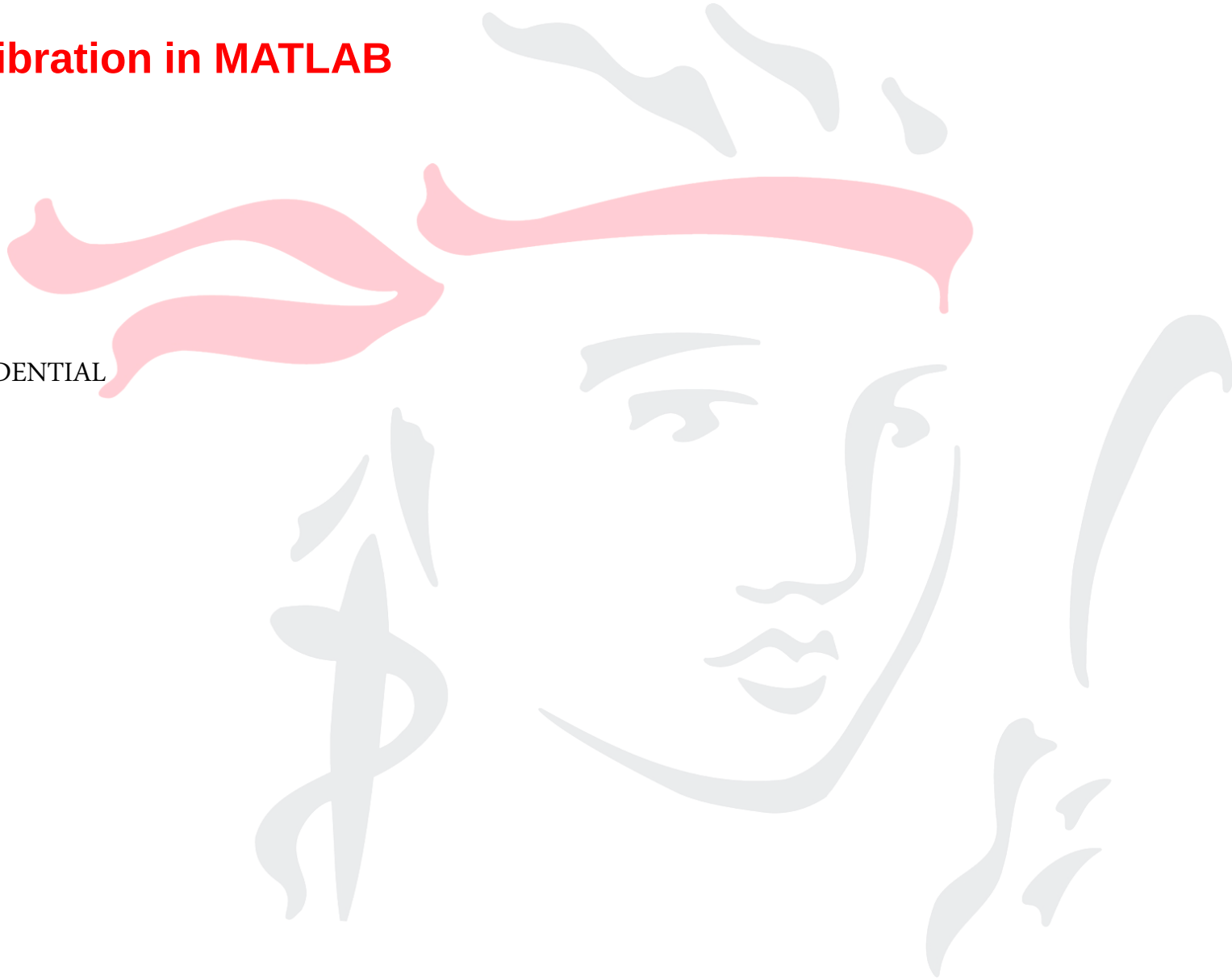
Model Calibration in MATLAB

Sam Bailey, PRUDENTIAL

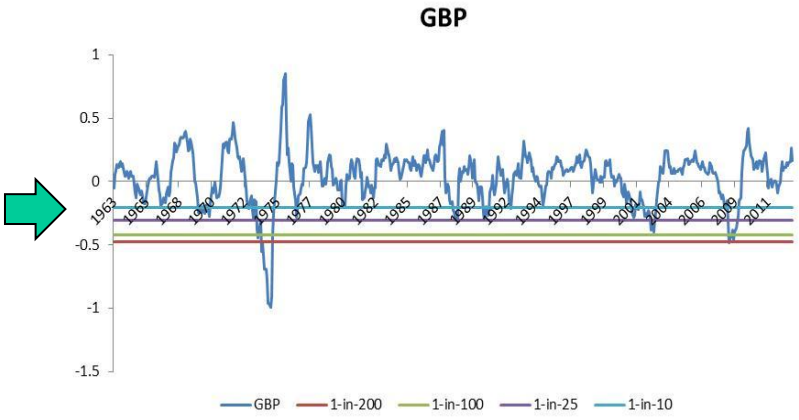


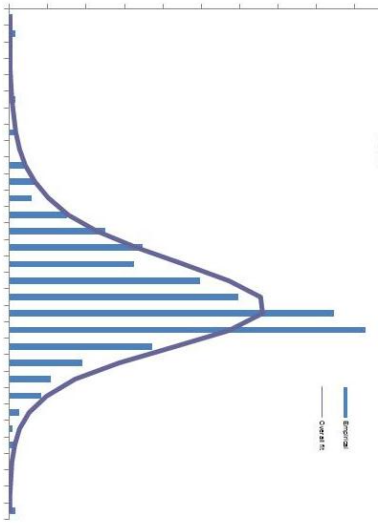
PRUDENTIAL

Calibrating the Risk Scenarios

Need to calibrate statistical models for all the market risks we are exposed to – for example

- equity level
- equity volatility
- interest rate level
- interest rate volatility
- credit spreads
- defaults
- property
- FX





Collect market data

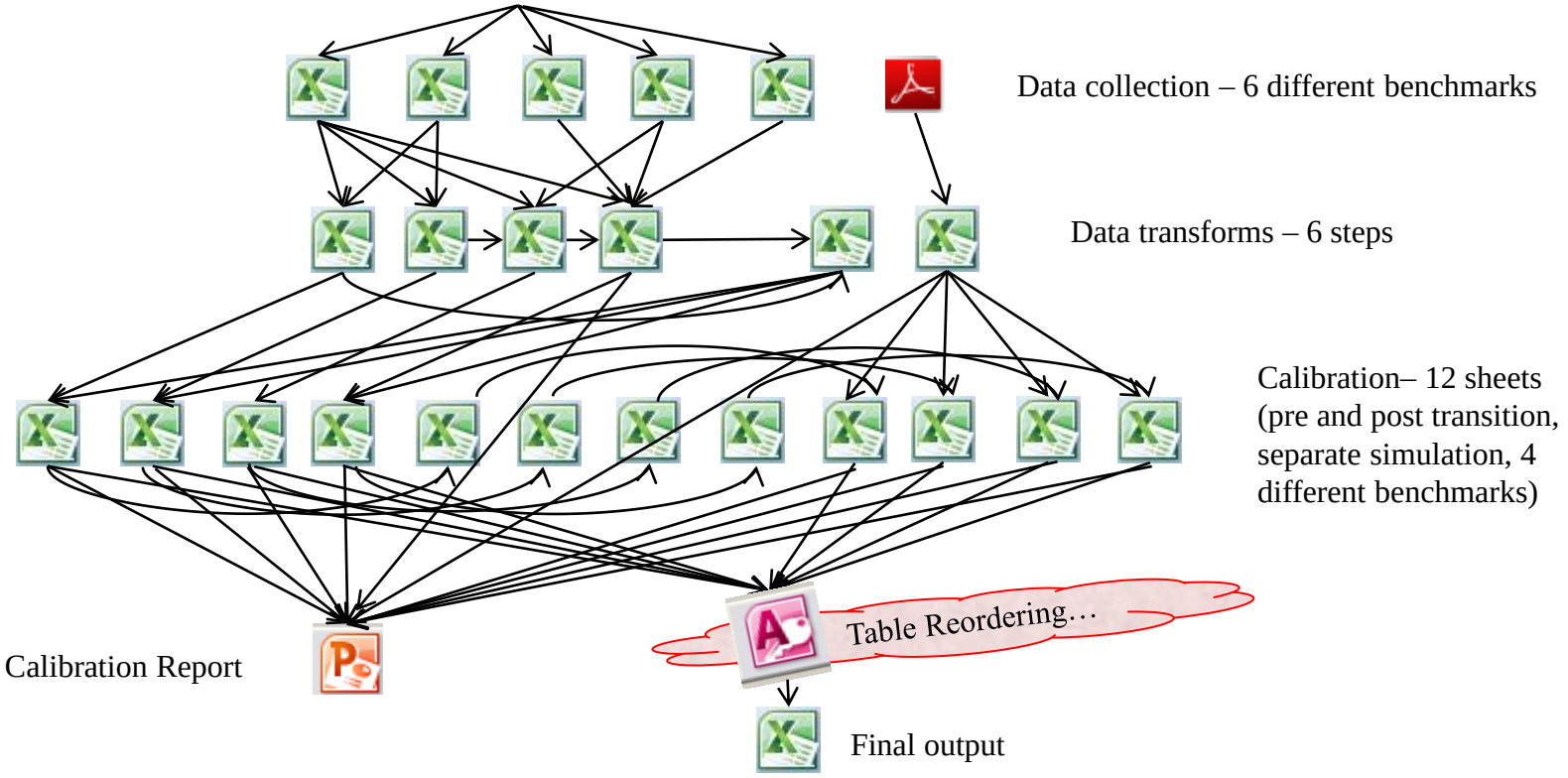
Transform

Statistical Fit

Current Calibration Process

Previously all done in spreadsheets

Worst case example – credit spread has total of 24 spreadsheets



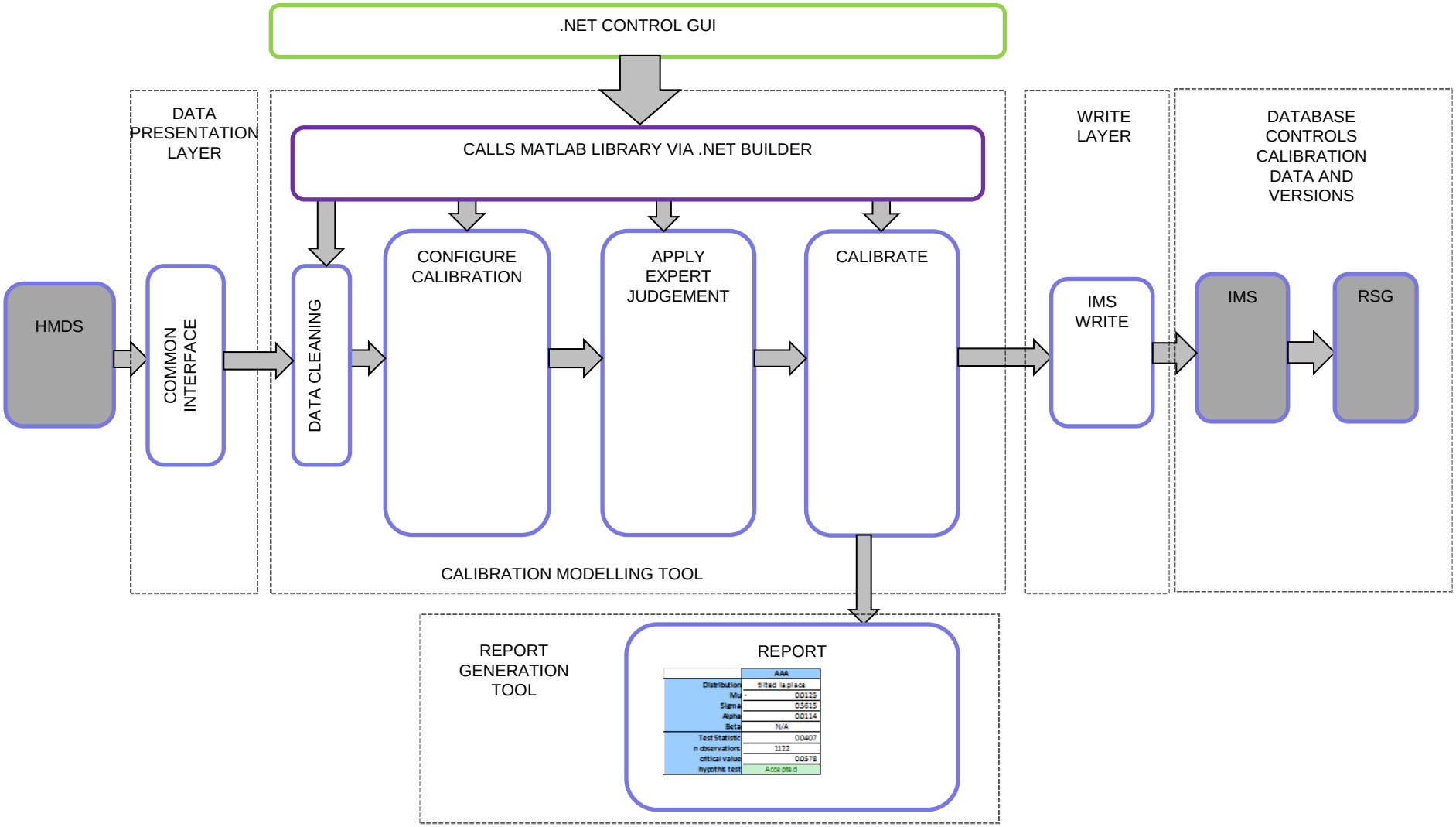
Approximately 50 dependencies, several hundred links and copy/pastes

Key Issues

- Very error prone
  - Enormous number of manual steps
  - Large number of copy pastes. Large spreadsheets prone to crashing.

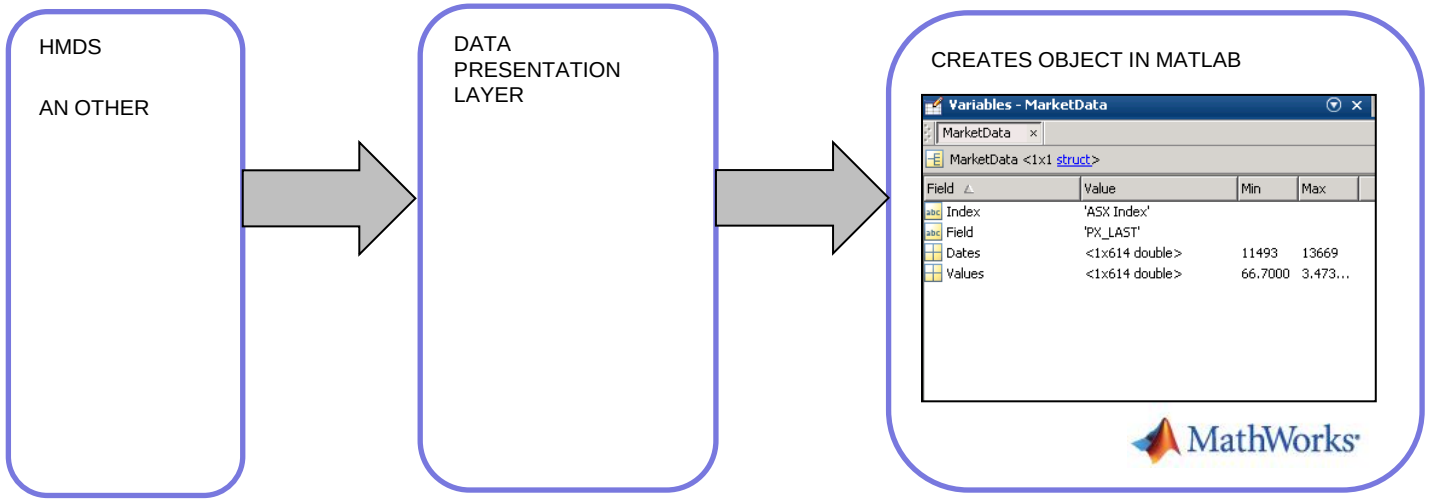
- Very time consuming
  - Approx 2 man months to run process end to end

Solution - Create a unified calibration system in MATLAB and .NET



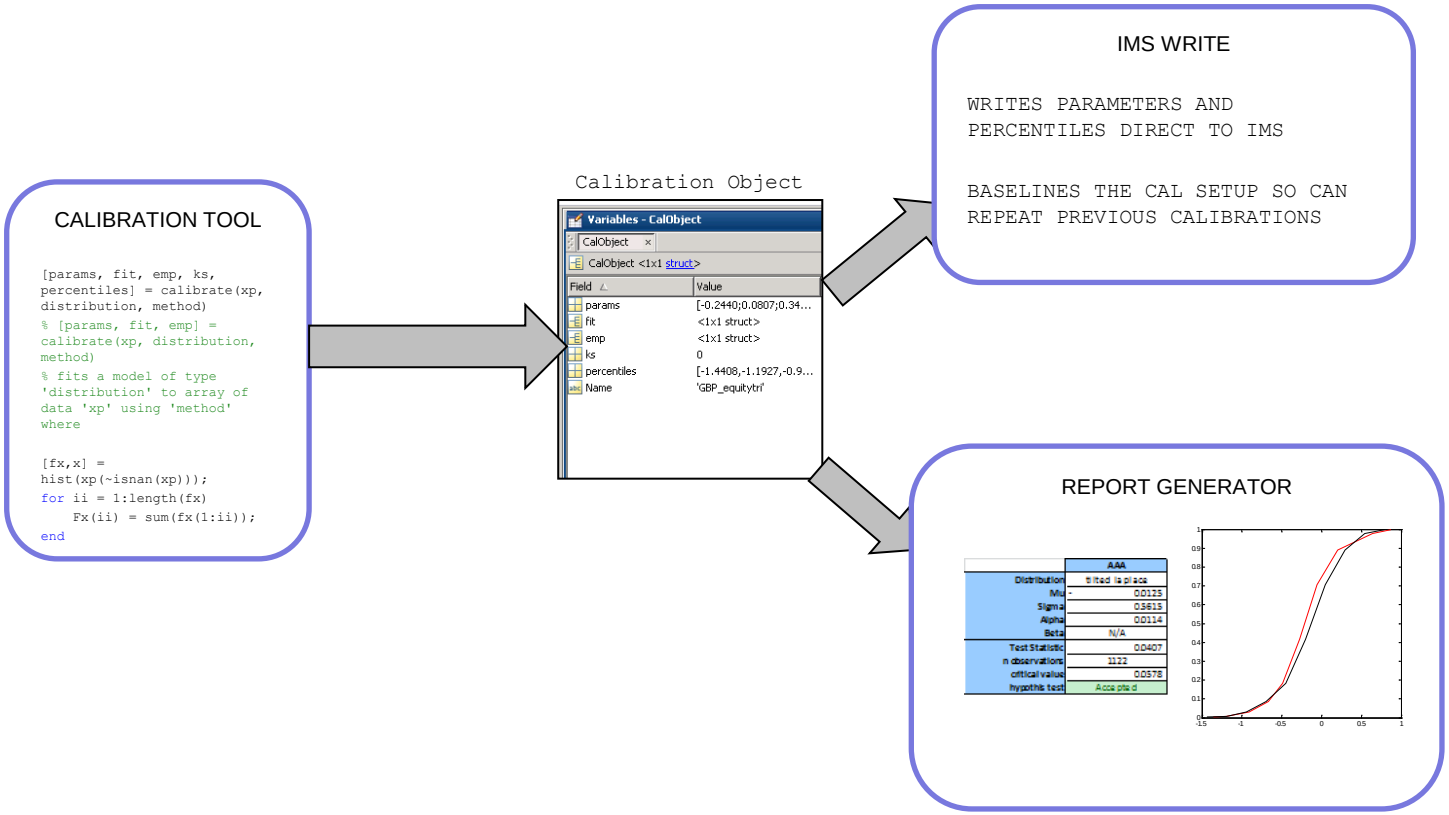
Data Presentation Layer

- Queries data from multiple sources (Market data database, Other e.g. spreadsheet)
- Creates a standard data object in MATLAB that the Calibration Tool can access



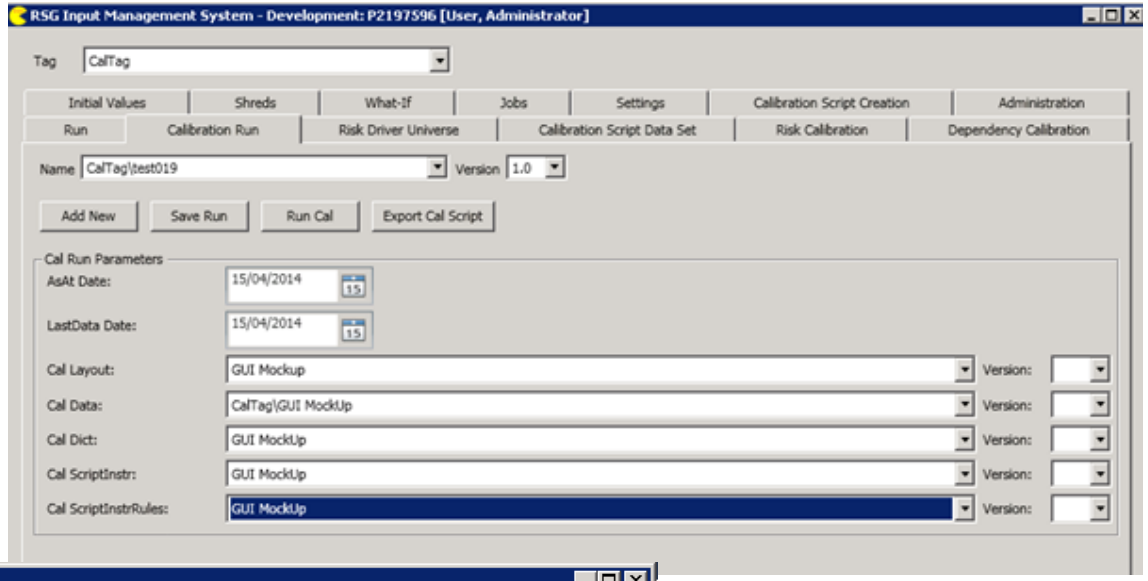
Data Upload to Input Management System and Report Generator

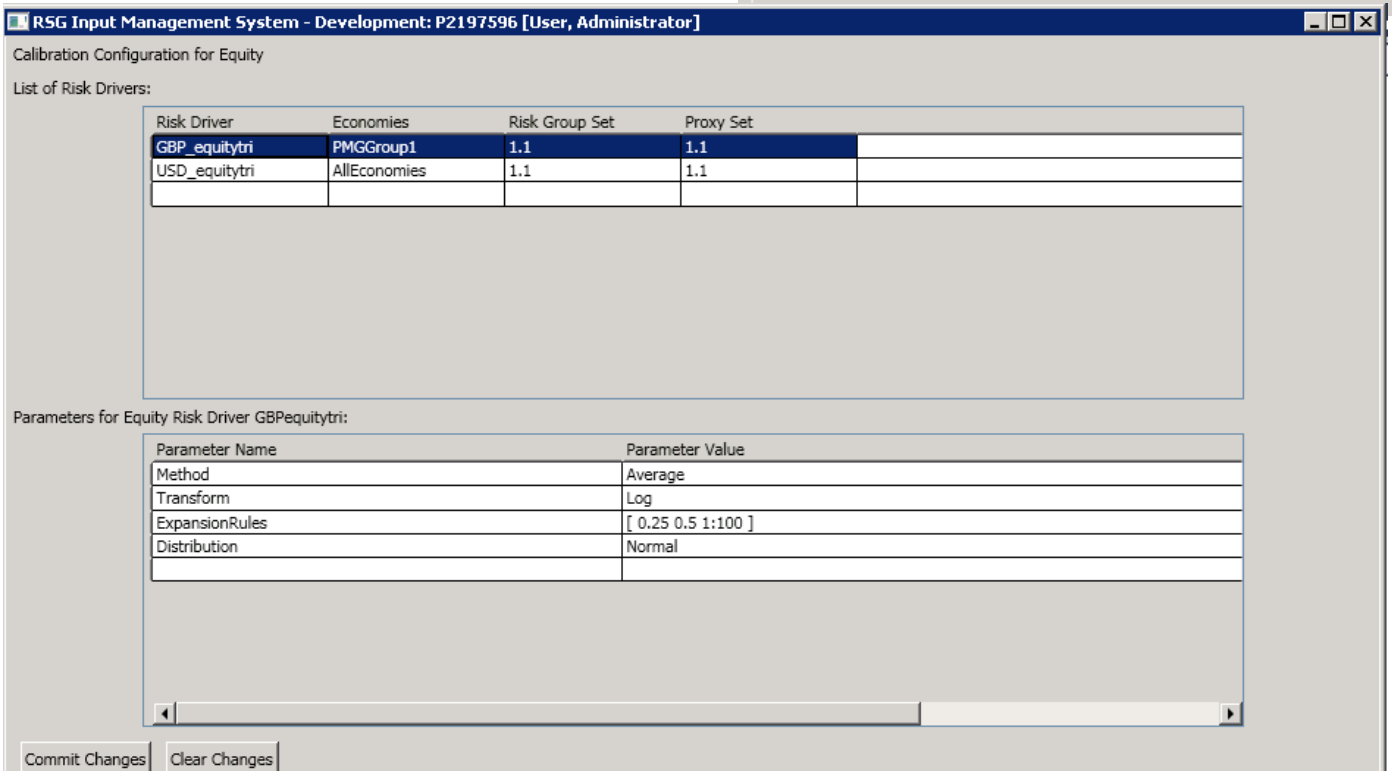
- Calibration Tool Outputs a Standard Calibration Object
  - Calibration Parameters
  - Distribution Percentiles
  - Fit Statistics (K-S test, Stationarity Test)



Calibration Interface

- Configures Risk (GBP - > FTSE)
- Version Controls the Calibration
- Stores for Posterity – can reproduce any calibration





Generates calls to MATLAB via .NET builder

MATLAB Builder NE for Microsoft .NET Framework

```
% CUR_equitytri = cal.equityCalibrate(RISK DRIVER, METHOD, TRANSFROM, PMGRISKPREMIUM, FIT, SKEWNESS AND KURTOSIS
SOURCE (OPTIONAL))

GBP_equitytri = cal.equityCal('GBP_equitytri','Average','log',3,'Normal',PMGGroup1);

USD_equitytri = cal.equityCal('USD_equitytri','Concatanation','log',3.25,'BestFit',AllEconomies);

CNY_equitytri = cal.equityCal('TWD_equitytri','Concatanation','log',3,'BestFit',AllEconomies);

VND_equitytri = cal.equityCal(averageReturns({'TWD_equitytri','MYR Index'}),'Individual','log',3,'BestFit');

THB_equitytri = forceFit(cal.equityCal('THB_equitytri','Individual','log',3,'BestFit'), 0.995, 0.45);

MYR_equitytri = scaleFit(cal.equityCal('MYR_equitytri','Individual','log',3,'BestFit'),1.5);

% writeToIms(CAL OBJECT, CAL NAME, CAL SET)

writeToIms(CNY_equitytri, 'FY14','1.1');
writeToIms(VND_equitytri, 'FY14','1.1');
writeToIms(THB_equitytri, 'FY14','1.1');
writeToIms(MYR_equitytri, 'FY14','1.1');

% writeReport(CAL OBJECT, CAL NAME, CAL SET)

writeReport(GBP_equitytri, 'FY14','1.1');
writeReport(USD_equitytri, 'FY14','1.1');
writeReport(CNY_equitytri, 'FY14','1.1');
writeReport(VND_equitytri, 'FY14','1.1');
writeTable({VND_equitytri.percentiles, CNY_equitytri.percentiles, USD_equitytri.percentiles,
GBP_equitytri.percentiles},'FY14','1.1');
```

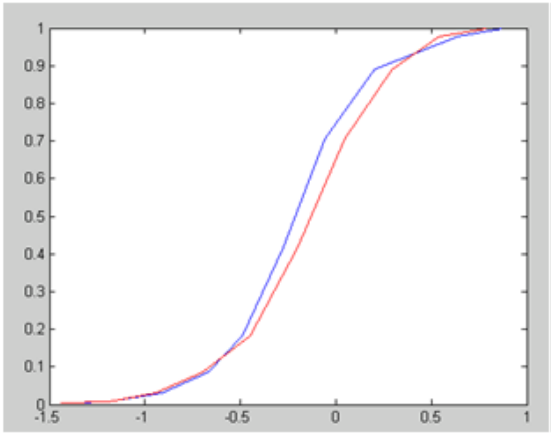
Can call from .NET in production, or make the same calls from MATLAB IDE during development

Automated Calibration Report Generation

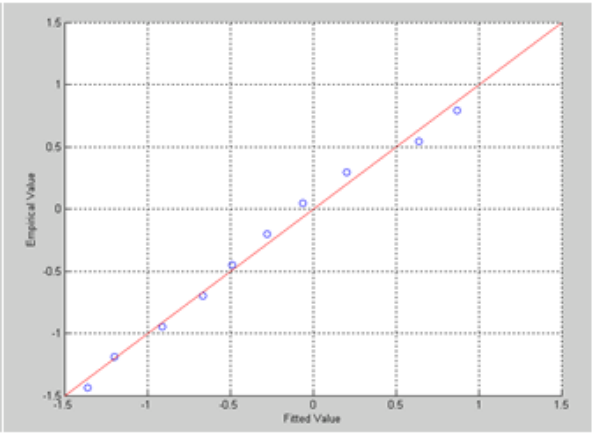
Calibration report for GBP_equitytri

Source Data 'ASX Index PX Last', 'ASX Index EQY DVD 12M'

CDF



QQ plot



Percentiles

Individual Tes	0.02%	0.05%	0.50%	1.00%	4.00%	5.00%	10.00%	50.00%	90.00%	95.00%	96.00%	99.00%	99.50%	99.95%	99.98%
GBP_equitytri	-1.441	-1.193	-1.057	-0.951	-0.965	-0.696	-0.662	-0.529	-0.445	-0.2	0.0475	0.2959	0.544	0.7921	0.834

Goodness of Fit Tests

Individual Tests	GBP_equitytri
KPSS_a	Pass
KPSS_l	Pass
DF	Pass
ADF_a	Pass
ADF_l	Pass
PP_a	Pass
PP_l	Pass
White	Fail
ANOVA	Pass
BF	Fail

Parameters

Parameter	Value
mu	-0.244
sigma	0.08072
alpha	0.34874
beta	0.30565

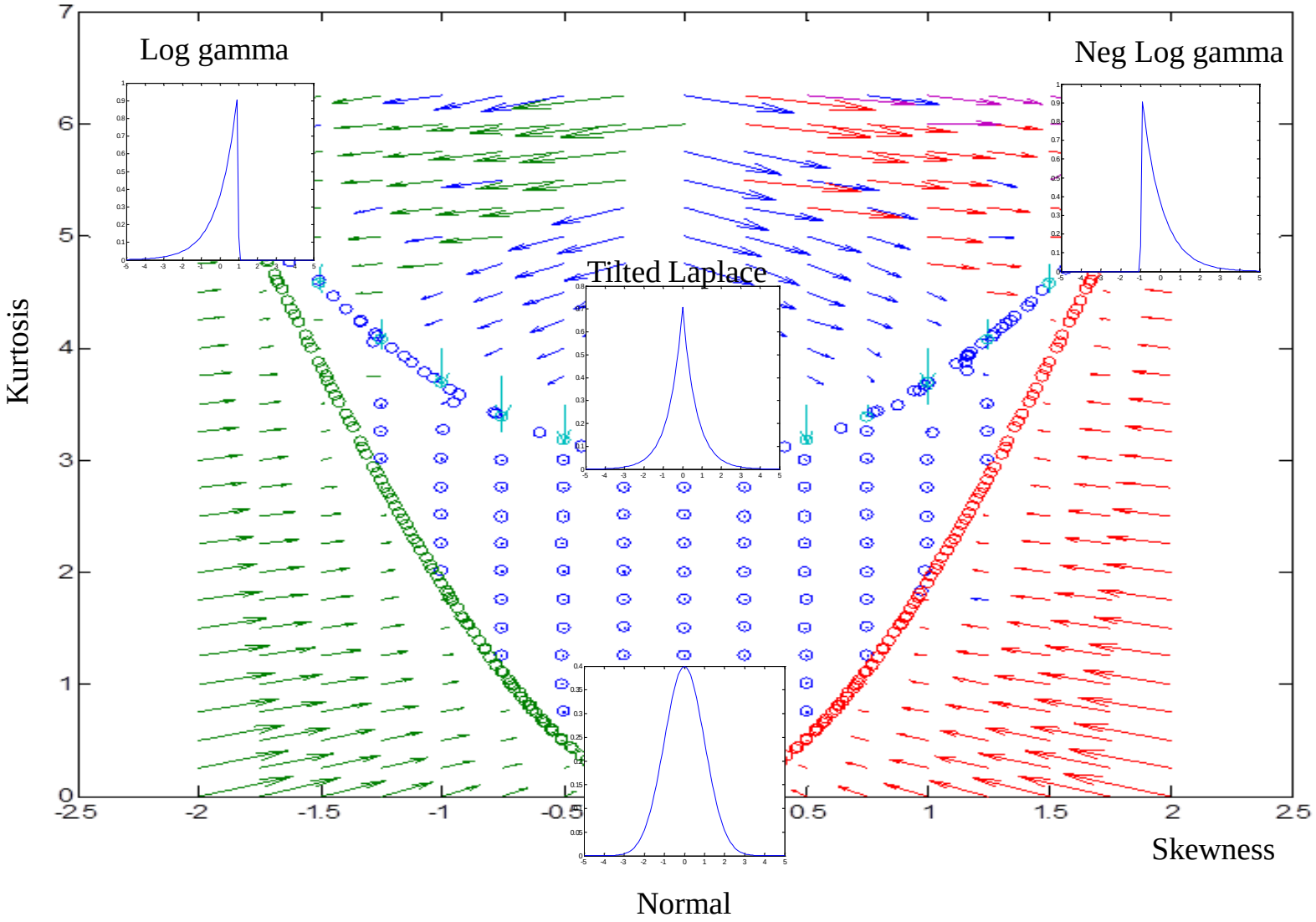
Fit Tool Suite – optimisation toolbox

- We use several distributions not in the Statistics Toolbox (EGB2, Tilted Laplace, Log Gamma, Negative Log Gamma)
- Full set of fitting tools and options
- Can be run in Object Oriented mode or as simple functions

```
classdef fitObj<handle

    properties
        params=[];
        distribution='';
        success=[];
    end

    methods

        function fitDistribution(obj, distMoments, varargin) ...

        function mymoments = analyticMoments(obj) ...

        function result = mypdf(obj, x) ...

        function result = myicdf(obj, x) ...

    end

end
```

Fit Tool Suite

- Using fmincon to minimise the errors in the Skewness and Kurtosis
- Can optimise over 4 parameters – previous tool (in Excel) could only do 1.



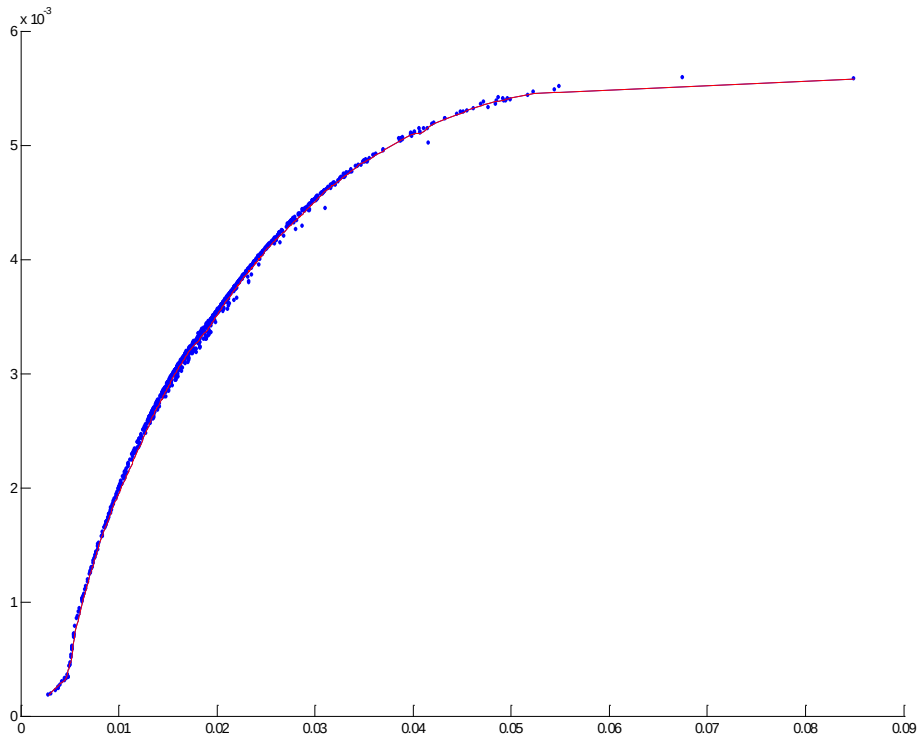
Solving cashflow equations – optimisation toolbox

- Determining the 'Cost of Downgrade' – cannot model analytically.
- Hold a bond portfolio for N years, with constraint that it must be investment grade (BBB or above).
- Some % of portfolio will end up below investment grade due to downgrades.
- Need to calculate the cost incurred to rebalance *as an additional spread*.
- Can project forward in time, problem becomes a form of the cashflow equation

$$DCF = \frac{CF_1}{(1+r)^1} + \frac{CF_2}{(1+r)^2} + \cdots + \frac{CF_n}{(1+r)^n}$$

- Cashflow equations cannot be solved analytically for interest rate, r. Need to call fmincon 100,000 times for our 100,000 stochastic simulations.
- Takes 45 minutes to run.

Using fminunc to solve cashflow equations



Solutions from fminunc for the interest rate are noisy (blue dots)

- Matches the Excel goal seek though.

Rather than solving for r for every simulation

- pre solve for fixed percentiles of the distribution.
- then interpolate.

Interpolation can be vectorised, whole model runs in 1.5 secs.

Conclusions

- We have migrated a time-consuming, error prone manual process in Excel into an automated tool in MATLAB and .NET
- Runs in a fraction of the time (press one button and wait 3 mins versus 2 months of copy/paste/refresh)
- Separates code from data
  - once we've tested the code base, we're confident it will work with the next version of the data
  - all vectorised – we never need to worry about whether there is 12 months of data or 90 years of data
- Use of the optimisation toolbox made the process quicker and easier
- Auditable, traceable, repeatable
- Easy to change settings and re-run