# Optimization under Uncertainty using GAMS: Success Stories and some Frustrations

Franz Nelißen

GAMS Software GmbH

FNelissen@gams.com

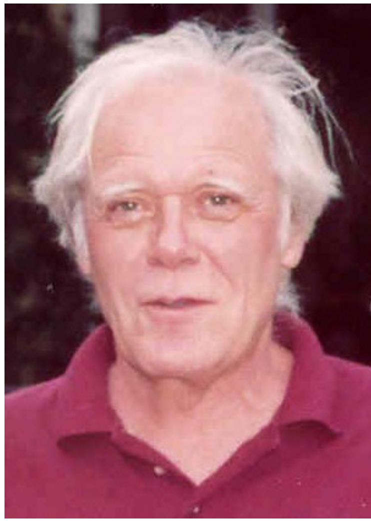
GOR Workshop “Optimization under Uncertainty ”

Bad Honnef, Germany, October 20-21, 2005

## Overview

- Algebraic Modeling Languages and GAMS
- Dealing with uncertainty
    - Static Models: The Mean Variance Model
    - Dynamic Models (Stochastic Programming)
- Summary

# GAMS Development Corp. & GAMS Software GmbH



- **G**eneral **A**lgebraic **M**odeling **S**ystem
- Started as a **research project** at the World Bank 1976
- Went commercial in 1987
- Professional **software tool provider**
- Operating in a **segmented niche market**
- Broad **academic** and **commercial** user base
- Offices in Washington, D.C and Cologne

# Modeling Tools

- Spreadsheets
- General programming languages
  - → C++, Delphi, FORTRAN, Java, VBA, …
- **Algebraic Modeling Languages**
- Specialized software for certain applications
- Mixture of different approaches

# What are Algebraic Modeling Languages?

→Allow **efficient handling** of **mathematical optimization problems**

Goals:

- Support the **decision making process**
- Increase **productivity** during model building and solution process
- **Adapt models** quickly to new situations

# Key Elements of Algebraic Modeling Languages

- **Declarative** approach
  - Implementation of the optimization problem is close to its mathematical formulation:
    - Variables, constraints with arbitrary names
    - Sets, indices, algebraic expressions, powerful sparse index and data handling
    - Efficient but simple syntax
  - Model formulation contains no hints how to process the model → Algebraic Modeling Language translates this representation into another form suitable for the optimization algorithm
- Also **procedural elements**: Loops, procedures, macros, …

# GAMS Basic Principles

- Balanced **mix** of **declarative** and **procedural** approaches
- **Separation** of **model** and **data**: Core model is data independent and scalable
- **Separation** of **model** and **solution methods**:
  - Multiple model types: LP, MIP, NLP, QCP, MIQCP, MINLP, MCP…
  - Maintained links to commercial and research algorithms (open solver interface)
- **Separation** of **model** and **operating system**: Models are platform independent
- **Open architecture** and **interfaces** to other systems: GUI, Excel, database, programming languages etc.
- **Maintainable** models and **protection of investments** in models

# Typical GAMS Application Areas*

- Agricultural Economics
- Applied General Equilibrium
- Chemical Engineering
- Economic Development
- Econometrics
- Energy
- Environmental Economics
- Engineering
- Finance
- Forestry
- International Trade
- Logistics
- Macro Economics
- Military
- Management Science and OR
- Mathematics
- Micro Economics
- Physics

* Illustrative examples in the GAMS Model Library

# Uncertainty in Finance

- Very **active** area with **significant contributions** to modeling and with important practical applications
- Some of the reasons:
  - Obvious impact of uncertainty
  - Dealing with uncertainty = *Risk Management* (**Basel II**)
  - *Real money" - a small difference may mean a big advantage*
  - High availability of data
  - Very competitive and liquid markets → Many instruments and strategies

# Optimization Models in Finance

- "Static" models: The **decision is made once**, **no further changes** possible
  - Mean-Variance Models
  - *Portfolio Models for Fixed Income*
  - *Indexation Models ("Tracking Models")*

  - *Scenario based Optimization*

- "Dynamic" models: **Sequence of decisions**
  - Stochastic Programming

# Mean-Variance Model

Markowitz (1952)→Nobel prize 1990

Some investments $x_i$ with historical data:

- Expected returns of investments: $\mu_i$: **Mean** of historical returns
- Risk: **Variance** of investments $Q_{i,j}$

Goal: **Balance risk** *r* of portfolio against expected **returns** of portfolio

→**Minimize variance** *v* of portfolio for a **given target return** *r*

## Mean-Variance Model Equations

$$Min \sum_{i=1}^{I}\sum_{j=1}^{J} x_i\, Q_{i,j}\, x_j$$

$$s.t. \quad \sum_{i=1}^{I} \mu_i x_i = r$$

$$\sum_{i=1}^{I} x_i = 1, \qquad x_i \geq 0$$

Algebraic Representation:

```
zdef..     v =g= sum((i,j), x(i)*q(i,j)*x(j));
rdef..     r =e= sum(i, mu(i)*x(i));
budget..   sum(i, x(i)) =e= 1;
x.lo(i) = 0;     # no borrowing
```

# Mean-Variance Model
# Core GAMS Model

```
$eolcom #
Set     i    analyzed investments; alias (i,j) ;
Parameter  q(i,j) variance matrix;
Variables  v      variance of portfolio,
           r      expected return for the portfolio,
           x(i)   fraction of the portfolio that consists
   of i;
Equations zdef    variance of portfolio
          rdef    expected return of portfolio
          budget  budget constraint ;
zdef..    v =g= sum((i,j), x(i)*q(i,j)*x(j));
rdef..    r =e= sum(i, mu(i)*x(i));
budget..  sum(i, x(i)) =e= 1;
```

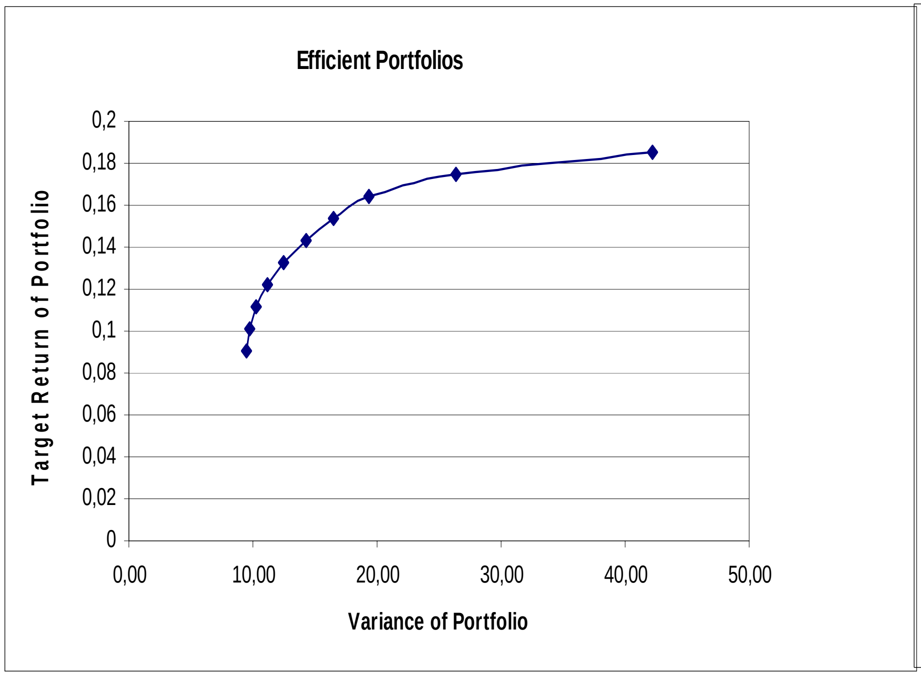
# Mean-Variance Model Data

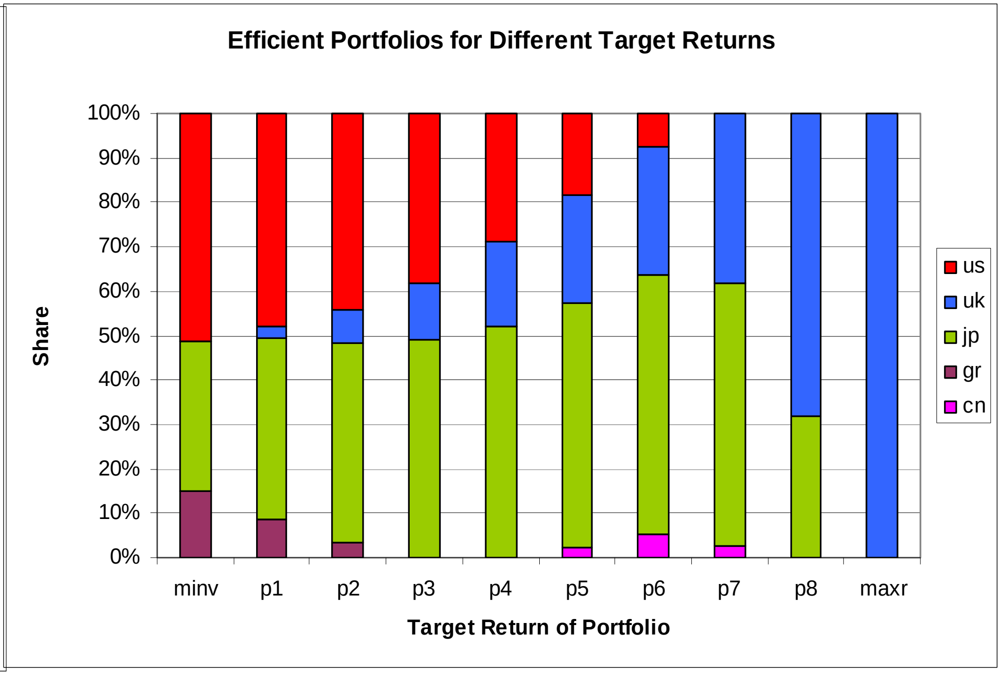
```
Set i / cn,fr,gr,jp,sw,uk,us    /;
Parameter  mu(i)/
 cn  0.1287,  fr   0.1096,  gr  0.0501,   jp      0.1524,
 sw  0.0763,  uk   0.1854,  us  0.0620                  /;
Table  q(i,j)
            cn       fr       gr       jp       sw       uk       us
    cn     42.18
    fr     20.18    70.89
    gr     10.88    21.58    25.51
    jp      5.30    15.41     9.60    22.33
    sw     12.32    23.24    22.63    10.32    30.01
    uk     23.84    23.80    13.22    10.46    16.36    42.23
    us     17.41    12.62     4.70     1.00     7.20     9.90    16.42 ;
q(i,j)$(ord(j) gt ord(i)) = q(j,i) ;
```

# Mean-Variance Model Procedural Elements

```
$include data.inc # get data from external file
x.lo(i) = 0;      # no borrowing
Model var / all / ;
set p     points for efficient frontier /minv, p1*p8, maxr/,
    pp(p) points used for loop           /      p1*p8      /;
parameter minr , maxr,            #minimal and maximal return
          rep(p,*), repx(p,i); # some quick reports
solve var minimizing v using qcp ;#find portfolio with minimal variance
rep('minv','return')   = r.l; minr=r.l;
rep('minv','variance') = v.l; repx('minv',i) = x.l(i);
solve var maximizing r using qcp ; # find portfolio with maximal return
rep('maxr','return')   = r.l; maxr=r.l;
rep('maxr','variance') = v.l; repx('maxr',i) = x.l(i);
loop(pp,   # trace efficient frontier
        r.fx = minr + (maxr-minr)/(card(pp)+1)*ord(pp);
        solve var minimizing v using qcp ;
        rep(pp,'return')   = r.l;
        rep(pp,'variance') = v.l;
        repx(pp,i)         = x.l(i);
     );
display rep, repx;
```

# Mean-Variance Model Solution





# Mean-Variance Model Extensions

- Different risk attitudes:

$$\min \; \lambda \sum_{i=1}^{I} \sum_{j=1}^{J} x_i \, Q_{i,j} \, x_j - (1-\lambda) \sum_{i=1}^{I} x_i \mu_i ; \; \lambda \in \{0, \ldots, 1\}$$

- $x_i$ may become negative → Allowing borrowing (Short Sales)
- Trading Restrictions ("Zero or Range" – Constraints) →Mixed Integer Quadratic Problem

# Trading Restrictions - Data

```
Table bdata(i,pd) portfolio data and trading restrictions
*                       - increase -           - decrease -
               old      umin    umax            lmin    lmax
        cn     0.2      0.03    0.11            0.03    0.11
        fr     0.2      0.04    0.10            0.04    0.10
        gr     0.0      0.04    0.07            0.04    0.07
        jp     0.0      0.03    0.11            0.03    0.11
        sw     0.2      0.03    0.20            0.03    0.20
        uk     0.2      0.03    0.10            0.03    0.10
        us     0.2      0.03    0.10            0.03    0.10
  ;
```

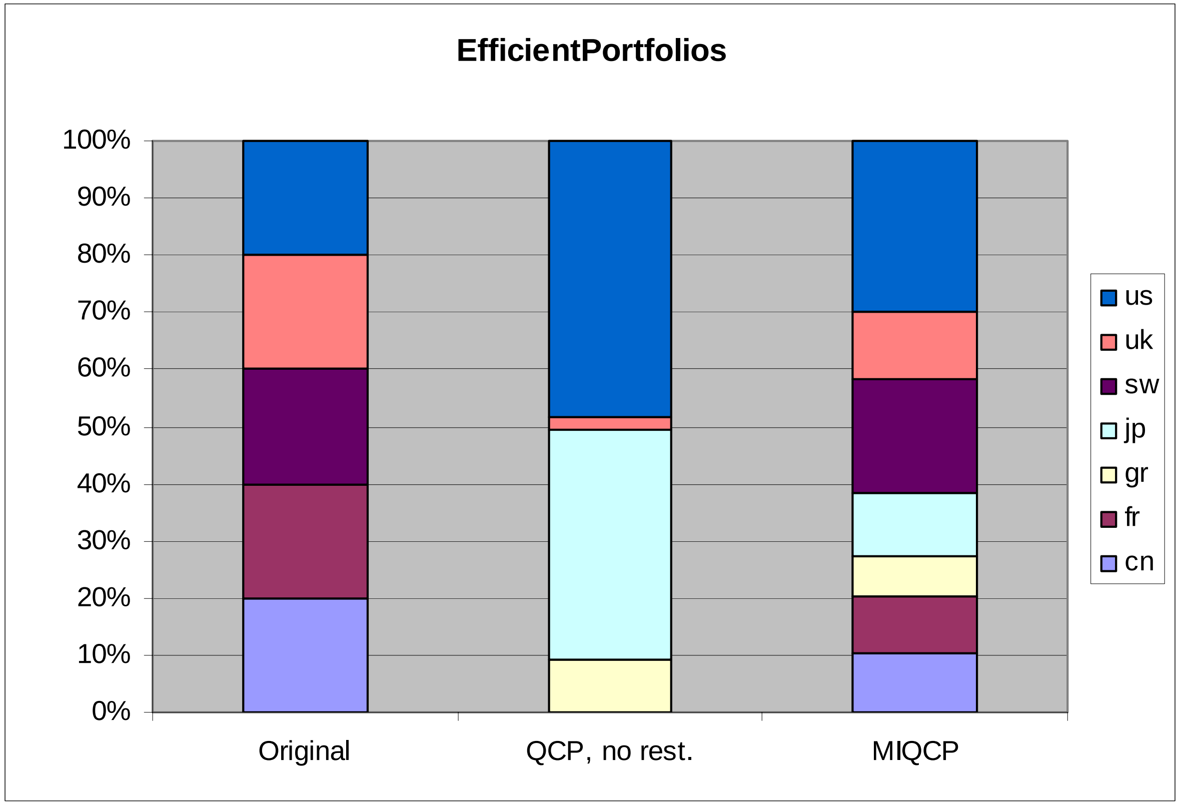
# Trading Restrictions – Formulation

```
Variables xi(i)     fraction of portfolio increase
          xd(i)     fraction of portfolio decrease
          y(i)      binary switch for increasing current holdings of i
          z(i)      binary switch for decreasing current holdings of i;
Binary Variables  y, z;
Positive variables xi, xd;
Equations  xdef(i)    final portfolio definition,
           maxinc(i)  bound of maximum lot increase of fraction of i,
           mininc(i)  bound of minimum lot increase of fraction of i,
           maxdec(i)  bound of maximum lot decrease of fraction of i,
           mindec(i)  bound of minimum lot decrease of fraction of i,
           binsum(i)  restrict use of binary variables;
xdef(i)..    x(i)   =e=  bdata(i,'old') - xd(i) + xi(i);
maxinc(i)..  xi(i)  =l=  bdata(i,'umax')* y(i);
mininc(i)..  xi(i)  =g=  bdata(i,'umin')* y(i);
maxdec(i)..  xd(i)  =l=  bdata(i,'lmax')* z(i);
mindec(i)..  xd(i)  =g=  bdata(i,'lmin')* z(i);
binsum(i)..  y(i) + z(i) =l= 1;
Model var2  /all/; Solve var2 minimizing v using miqcp ;
```

# Trading Restrictions – Solution

*Variance of Portfolio:*

- *Original:      21.419*
- *QCP, no Rest.: 9.66*
- *MIQCP:         14.853*



## Recent Developments

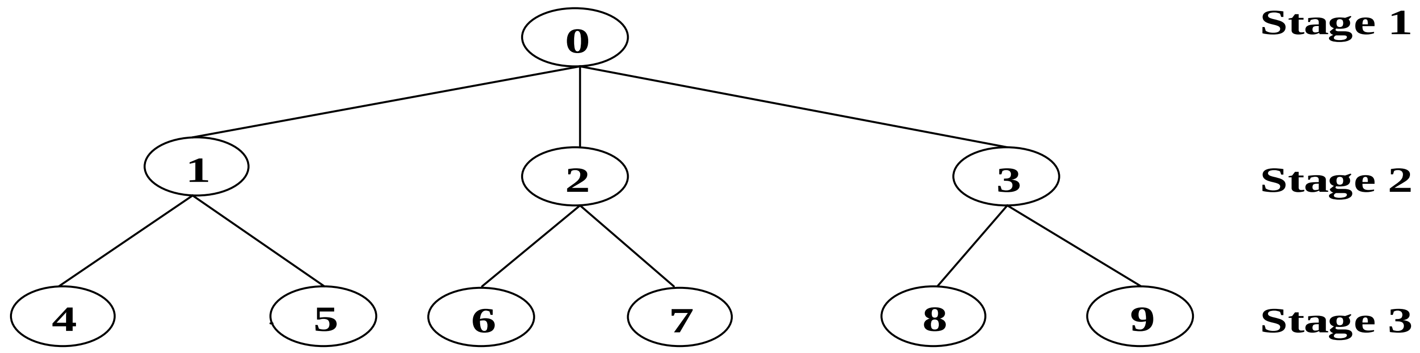
- Support of codes, which take advantage of special problem structures:
  - →Quadratic / Mixed Integer Quadratic Programming via Interior Point Methods / QP Simplex
- Support of global optimization codes

## Stochastic Programming

- Stochastic Programming models allow sequence of decisions
- Elements:
  - **Scenarios**:
    - Complete set of possible discrete realizations of the uncertain parameters with probabilities
    - Capture complex interactions between different uncertain parameters (risk factors)
    - What are “good scenarios”?
    - How many scenarios are necessary?
    - How do we generate scenarios?
  - **Stages**: Decisions points. First stage decisions now, second stage decision (depending of the outcome of the first stage decision) after a certain period and so on

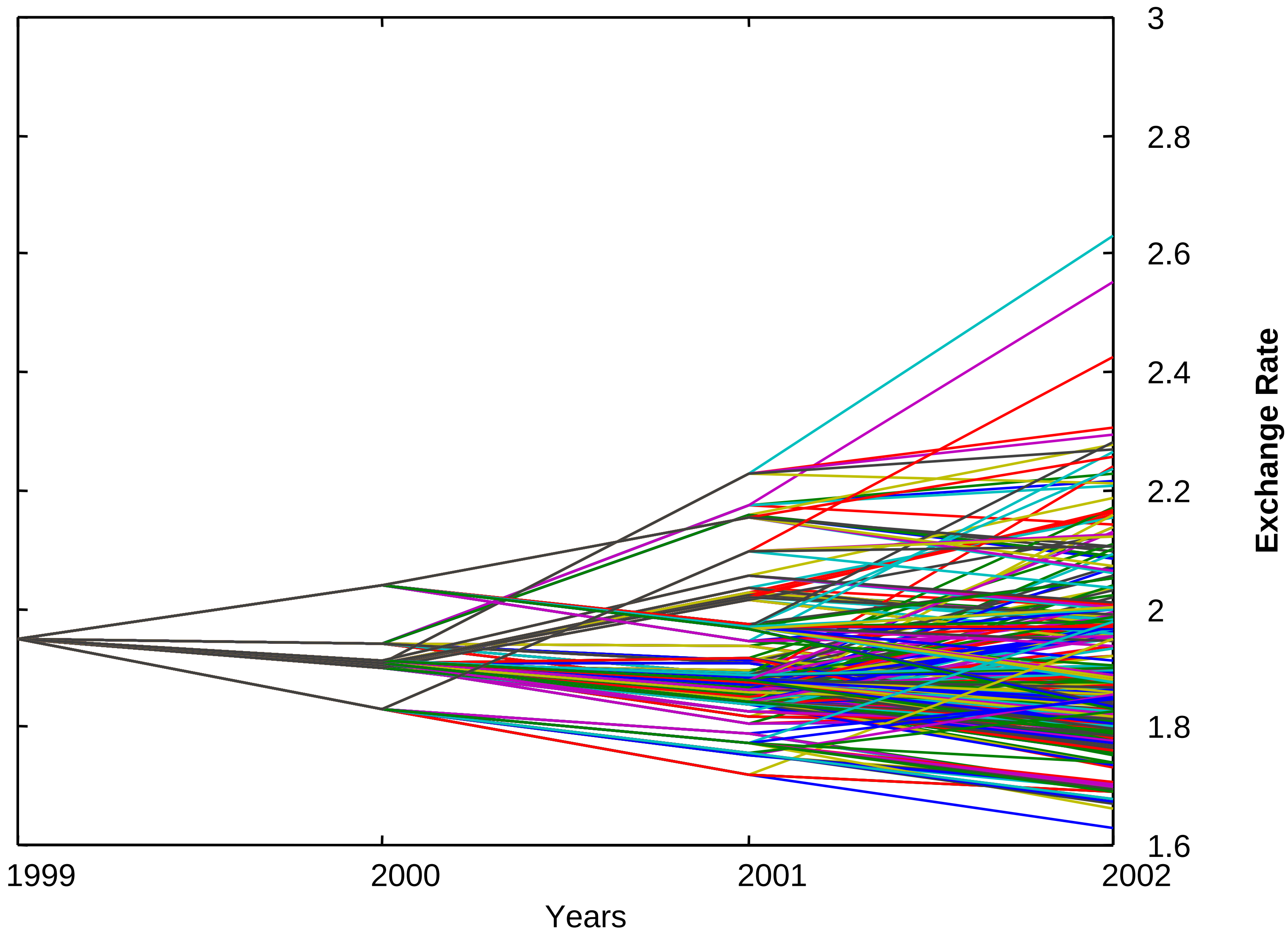
  →**Recourse**: Describes how **decision variables** can **adept** to the **different outcomes** of the random parameters at each stage

# A simple Scenario Tree



- combines scenarios, stages and probabilities

# Another Scenario Tree



# Stochastic Programming Some Challenges

- **Domain** specific knowledge
- **Impacts** of uncertainty:
  - Does it make a difference and is it worth the effort?
  - How far can one get with a certain budget?
- **Development** and **Fall-Back Strategy**?
- **Data** (availability and importance of certain and uncertain parameter) → Generation of “good” scenarios and definition of stages
- Interpretation and Presentation of **Results**
- **Maintenance** of the system

## Stochastic Programming Technical Challenges

- **Deterministic equivalent**: Includes all scenarios and stages → Size of model explodes
- Challenges (among others):
  - Programming and generation difficult
  - Solution may not be possible
  - Interpretation and validation of results

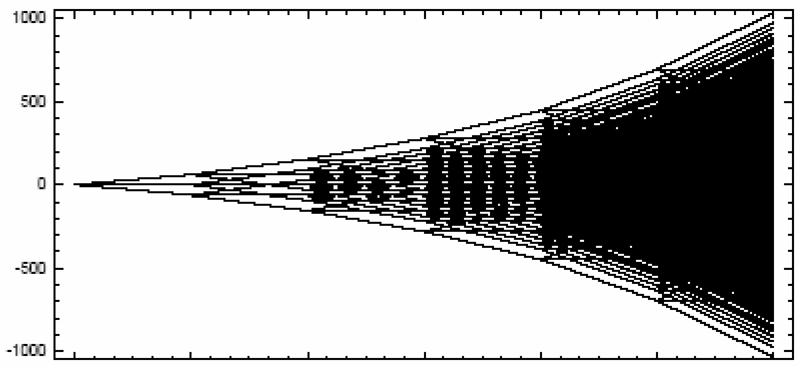
# Facing these (tech.) Challenges

1. How does GAMS support the modeling of Stochastic Programming Problems?
2. Some scenarios only differ slightly → Can we **reduce** the **number of scenarios**?
3. Stochastic Programming **Problems** are **structured** → How can we take advantage of **specialized solution techniques** for Stochastic Programming

# Stochastic Programming in GAMS

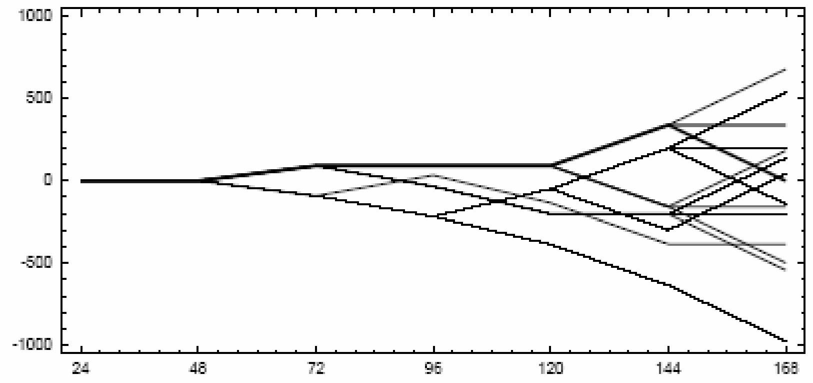
- Support for **huge problem instances**: **64 Bit** OS (PC) and **Grid Computing** (experimental)
- **Reliable** and **fast  import** and **export** of data and results
- **Visualization** of results
- **New language elements**  might improve reliability of :
  - Random distributions for some problem data
  - Special expressions and conventions for scenario trees and stages
    - Special sets for trees: Root, nodes, leafs,  ancestor and child relations; automated generation of trees
    - Connection of variables or constraints to certain stages: A variable or constraints is only active at a certain stage: x.stage(i)=1;

# Scenario Reduction

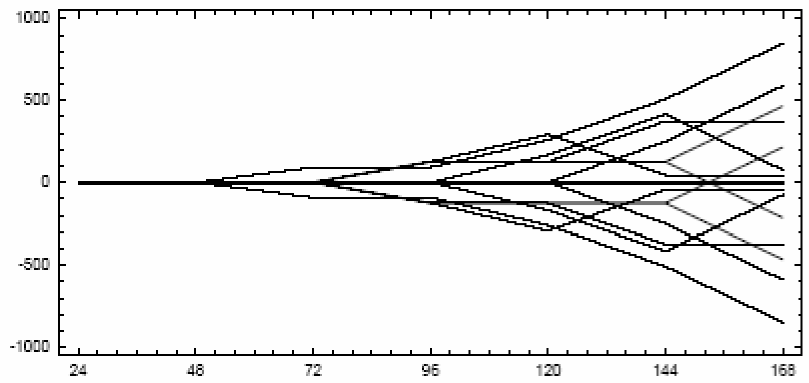


Original load scenario tree

→ *Goal: Find an* ***approximation of the original scenario*** *tree with* ***less nodes***



Reduced load scenario tree / backward



Reduced load scenario tree / forward

## Scenario Reduction Steps

1. Write a the stochastic model including the full tree structure
2. Pass the tree structure to SCENRED*
3. Reduce tree and reallocate probabilities
4. Import reduced  tree back to GAMS
5. Solve the model with the reduced tree structure

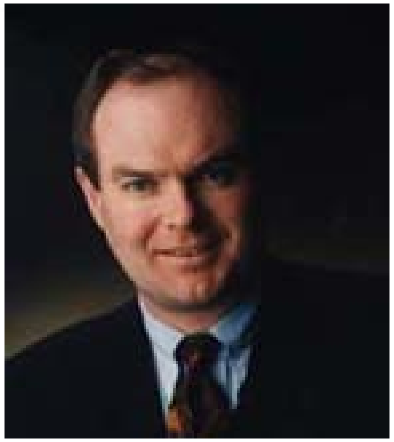
* SCENRED has been developed by Groewe-Kuska, Heitsch & Römisch, Humboldt-University Berlin, Germany

## Specialized Algorithms

OSL Stochastic Extensions (IBM):

- Solves **n-stage stochastic linear programs** with recourse
- Nested **Benders decomposition**
- Requires **deterministic equivalent** representation of the problem, which may be **huge** but is **solver independent**
- GAMS made substantial investment producing a solver independent interface, but unfortunately the product is **no longer supported** by IBM

# Specialized Algorithms



DECIS (Infanger)

- Solves **two-stage** stochastic linear programs with recourse
- **Benders decomposition** and advanced importance sampling techniques
- Requires additional information describing the uncertain elements of the core model in a form, which is **not compatible** with other solvers

## Specialized Algorithms

Main Problem:

→**No uniform (solver independent) problem representation** (both for the input and output)

→**Various approaches**, not yet clear which standard will be adopted

## Summary

- Large classes of problems can be solved
- GAMS provides a powerful and flexible framework for these classes of models

- Stochastic Programming still challenging and developing field
- Limited application of Stochastic Programming in practice