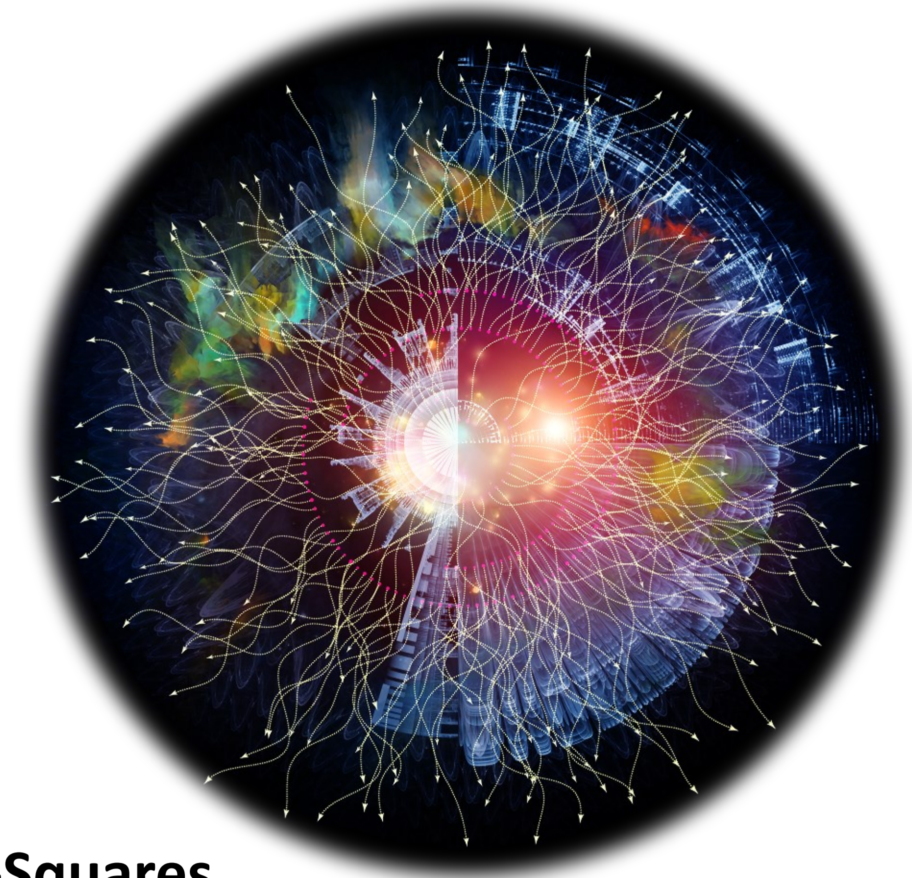
Deloitte.



Neural Networks Meet Least-Squares Monte Carlo at Internal Model Data

Bern, 6 September 2024

Zoran Nikolić

MAKING AN IMPACT THAT MATTERS since 1845

Speaker

Zoran Nikolić

Agenda

1.

Least-Squares Monte Carlo

- Internal Model within Solvency II framework
- Proxy modeling for SCR calculation

2.

Internal Model Data Published by DAV

- Data specifically generated for public use
- Description of the dataset

3.

Polynomials as Proxy Functions

- Forward step-wise adaptive algorithm
- Usage of AIC for term selection/regularization

4.

Neural Networks as Proxy Functions

- Feed-forward artificial neural networks
- Training using model data

1. Least-Squares Monte Carlo

Approximation of Own Funds (OF) in an Internal Model

SCR calculation with an OF proxy function

Risk factors

Changes in the chosen risk factors in a one-year time horizon (longevity, lapse, IR, equity, etc.). These serve as input variables or features for regression models

Real-World One-Year Projections

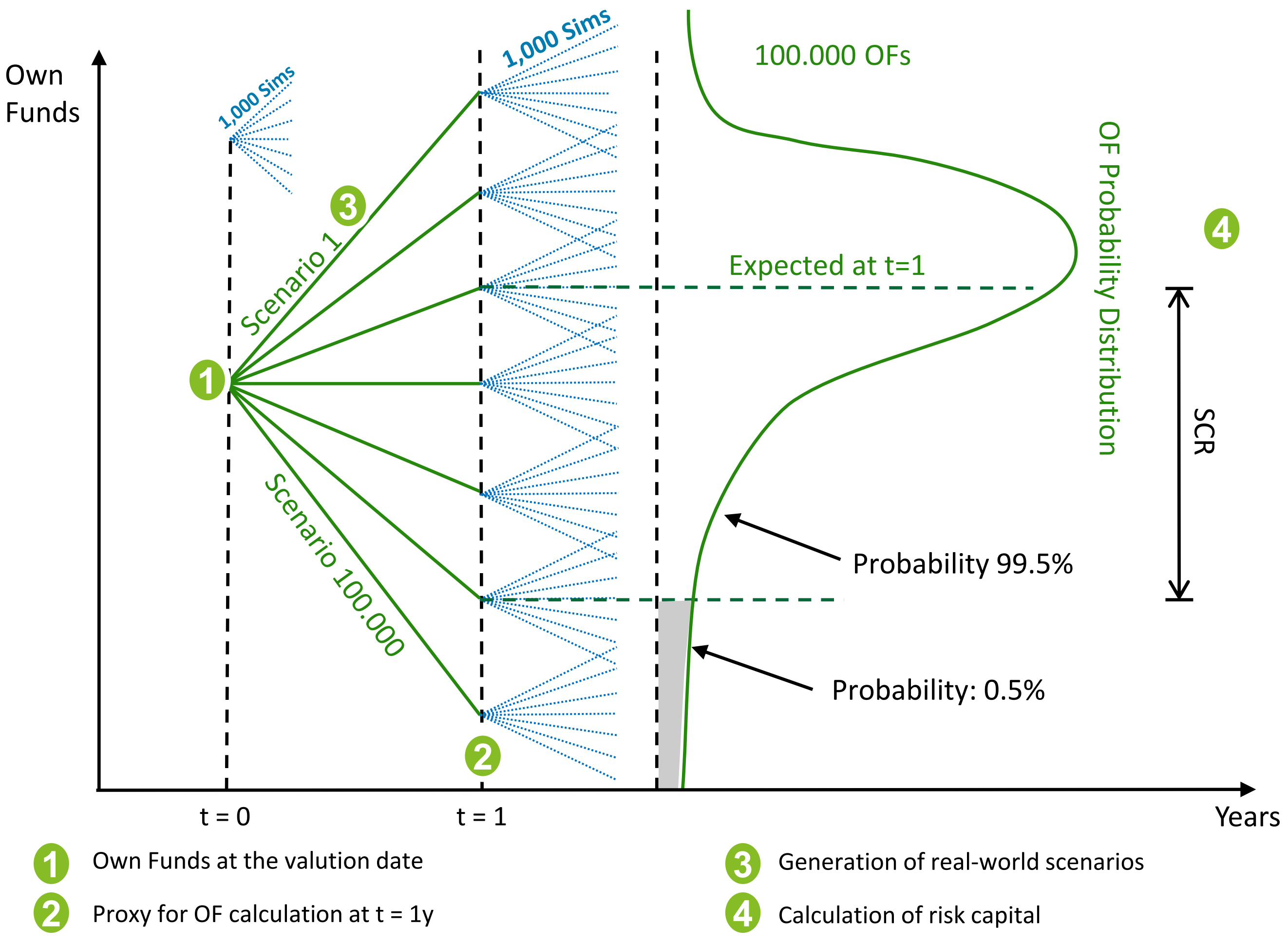
Sampling from a joint distribution of the risk factor changes over one year for a full empirical distribution of OF (e.g. 100,000 scenarios)

Monte Carlo Risk-Neutral Simulations

Monte Carlo simulations of "heavy" actuarial projection models (e.g. 1,000 simulations)

→ "Nested stochastic" problem with exploding computational requirements
→ Mitigation through proxy functions

SCR Calculation Workflow Using Own Funds (OF) Approximation

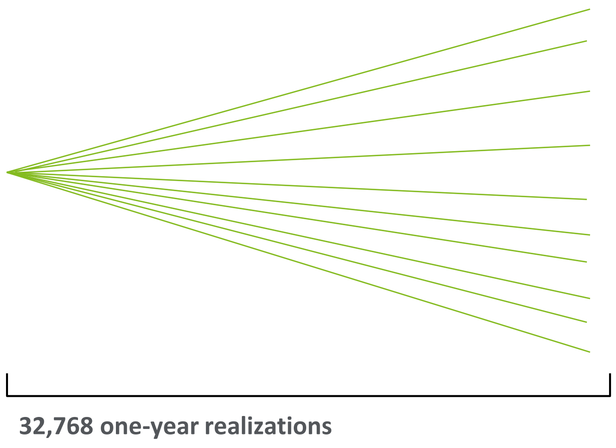


Least-Squares Monte Carlo Method

An efficient usage of the available scenario budget

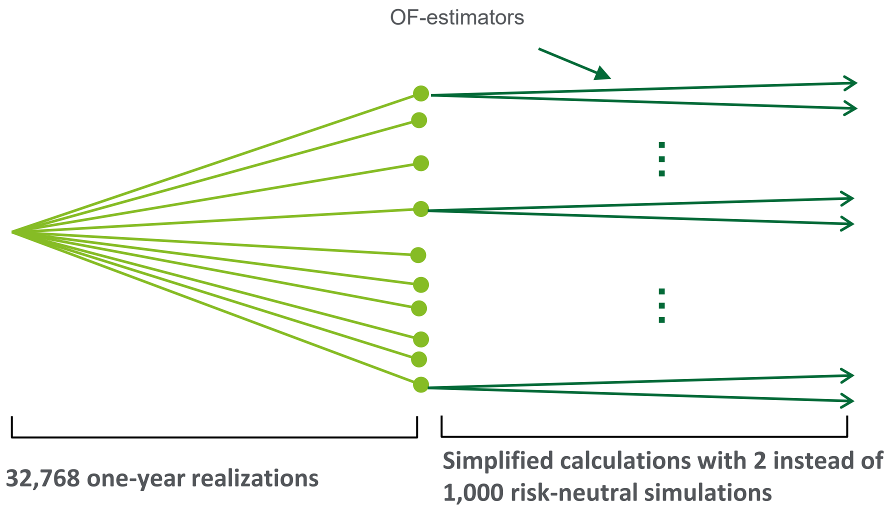
2a Regression Scenarios

- Generation of a smaller number **of scenarios (e.g. 32,678** in what follows later**)**
- Scenarios reflect possible one-year developments of **real-world risk drivers**
- Each scenario includes changes to **all** risk factors



2b Projection-model Runs

- For each scenario an inaccurate Monte Carlo valuation with **two instead of 1,000** risk-neutral simulations
- Hence calculation of **32,768 inaccurate OF estimators**

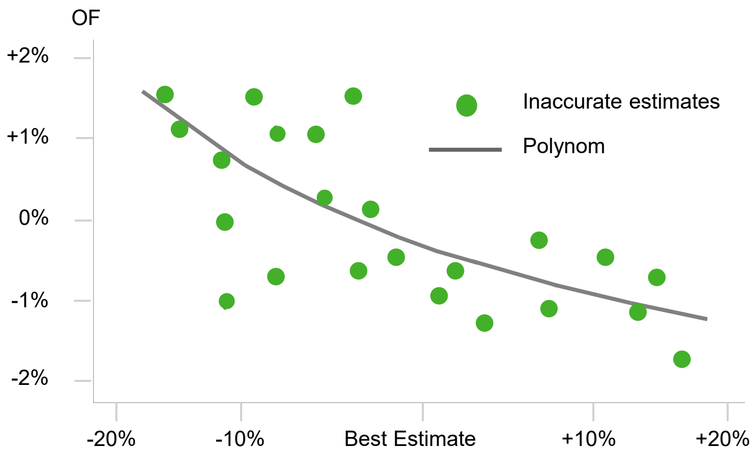


Least Squares Monte Carlo Method

Different regression functions possible

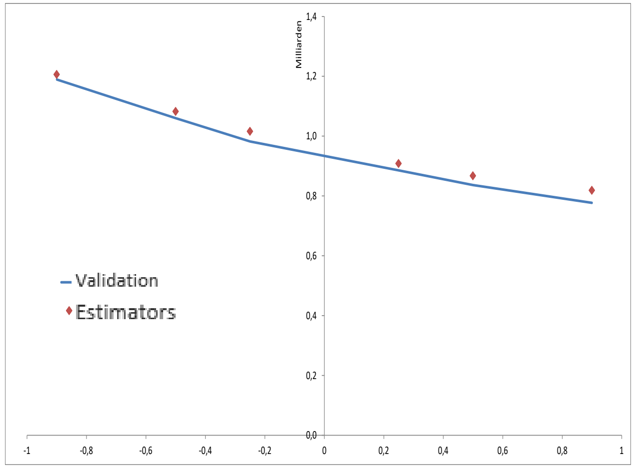
2c Regression

- Regression of a multidimensional function:
  - **Proxy-function** for the OF
- Risk drivers: e.g. costs in the image below



2d Validation

- Validation of the proxy function
  - **Out-of-sample** tests
  - Accurate with **1,000 or 4,000** risk-neutral simulations

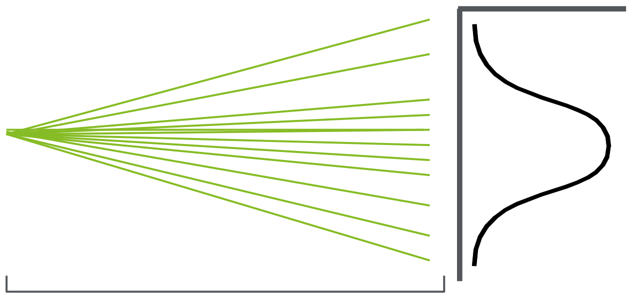


Least Squares Monte Carlo Method

Use LSMC proxy instead of the actuarial projection model

3 Risk Drivers Distribution

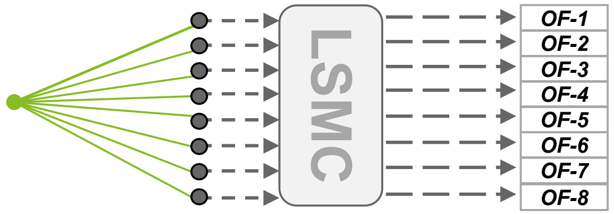
- Simulation of the joint risk driver distribution



100,000 one-year realizations

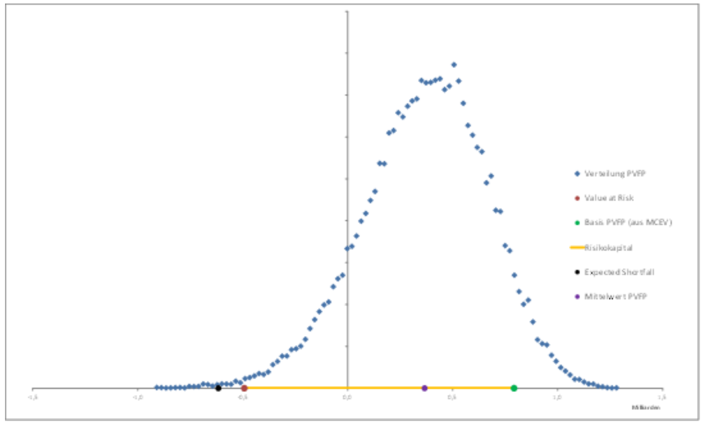
4a Evaluation

- Evaluation of the LSMC function



4b SCR and Own Funds

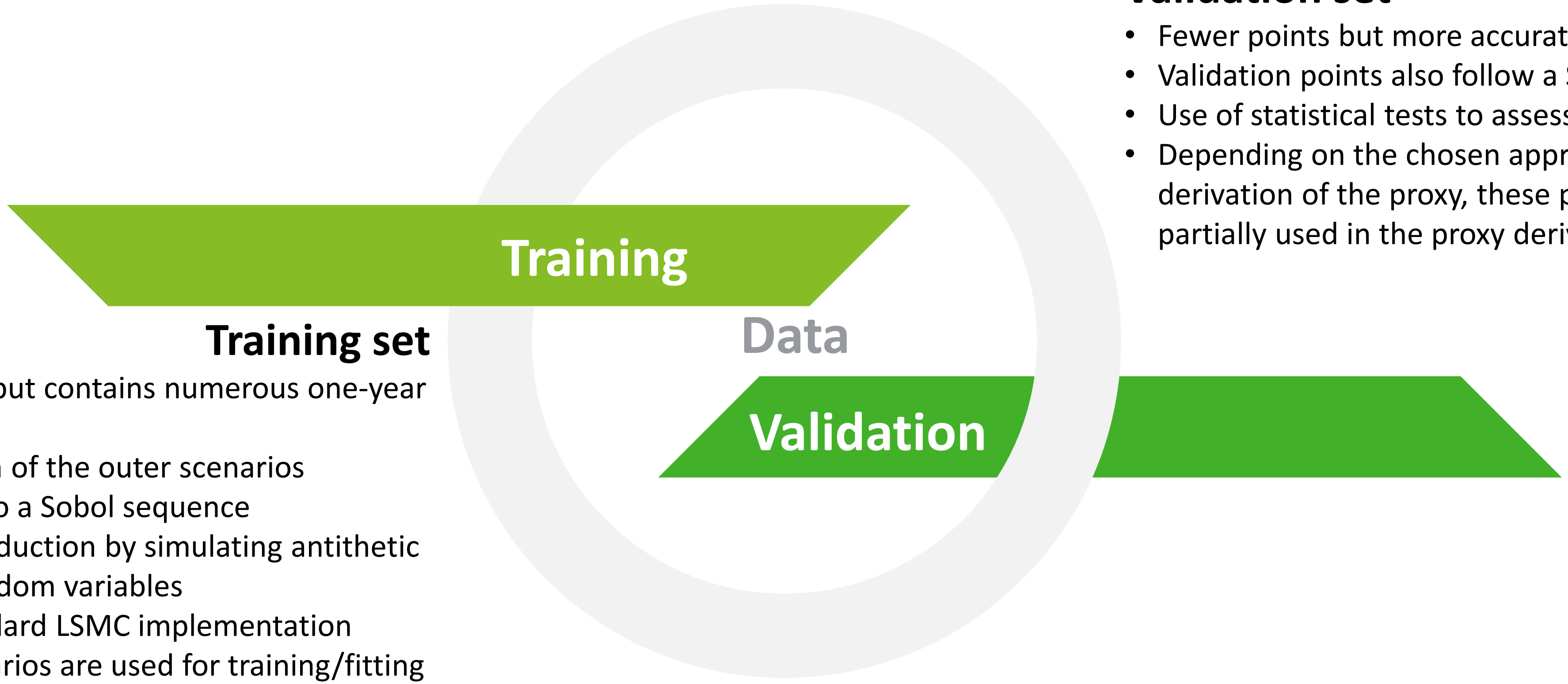
- Determination of Own Funds distribution and SCR determination



2. Internal Model Data Published by DAV

The Data Set For Three Real Anonymized Insurance Portfolios

Training and Validation Data



Validation set

- Fewer points but more accurate evaluations
- Validation points also follow a Sobol sequence
- Use of statistical tests to assess proxy goodness
- Depending on the chosen approach for the derivation of the proxy, these points can also be partially used in the proxy derivation process

Training set

- Imprecise, but contains numerous one-year scenarios
- Distribution of the outer scenarios according to a Sobol sequence
- Variance reduction by simulating antithetic pairs of random variables
- In the standard LSMC implementation these scenarios are used for training/fitting a proxy function describing the relationship between risk factors and Own Funds

The Data Set For Three Real Anonymized Insurance Portfolios

The Base Point and SCR Region Scenarios

Data

Base point

- Represents reference point in the calculation of the SCR
- No risk factor change
- Very accurate evaluation

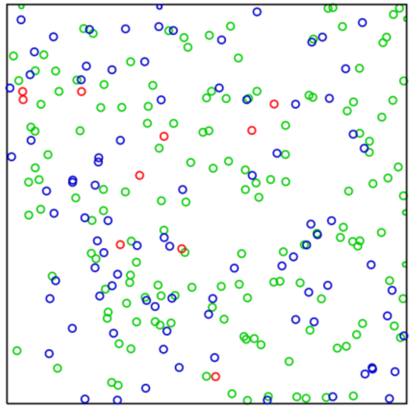
SCR Region

- The dataset contains additional “special” validation points for which the company is experiencing a loss close to the 0.5% loss which is used for SCR derivation
- Very accurate evaluation in order to reduce the Monte Carlo error as much as possible for these important scenarios
- This data set gives a flavor of the “nested stochastic” modeling

32,768 Training Scenarios as a Sobol-Sequence

An Alternative to Uniformly Distributed Random Numbers

Uniformly distributed random numbers *vs.* **Sobol-Sequence**



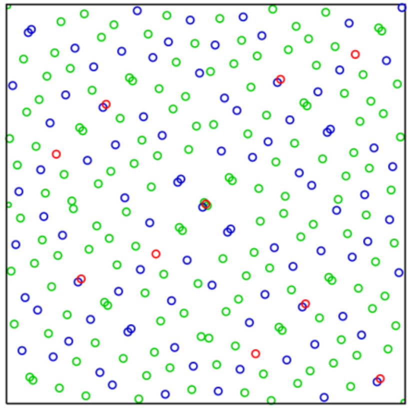


Image from: https://en.wikipedia.org/wiki/Sobol_sequence

Definition

A Sobol sequence is a quasi-random sequence

Properties

- Sobol sequences take values in $[0,1]^n$ with n number of risk factors
- Joint sampling of risk factors
- No dependency on the seed

Advantages

- Higher uniformity than ordinary generators of the uniform distribution
- Easy to add or remove points/dimensions
- Easy to analyze a subset
- Very flexible process

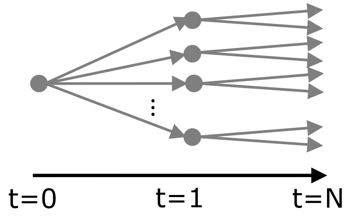
Bottom line

Sobol sequences not necessarily superior, but offer an efficient and flexible generation process

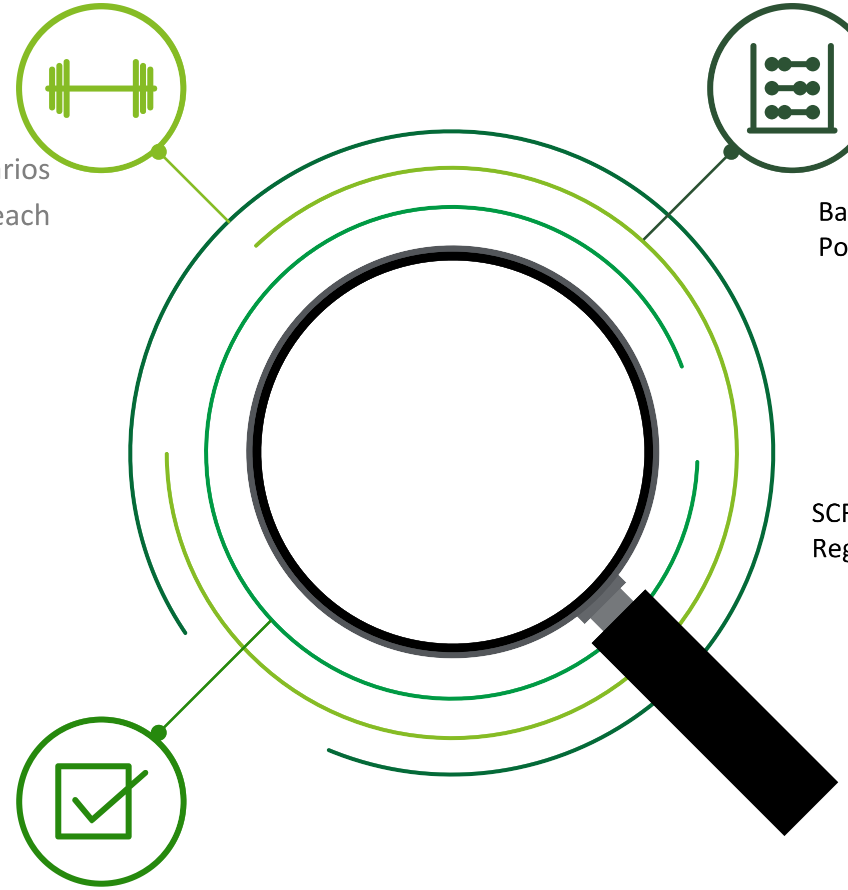
Number of Scenarios in the Data Sets Published by DAV

Various types of data sets for three portfolios

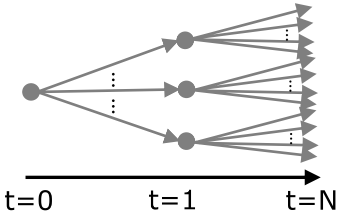
1. Data for Training



32,768 one-year scenarios with **two** simulations each

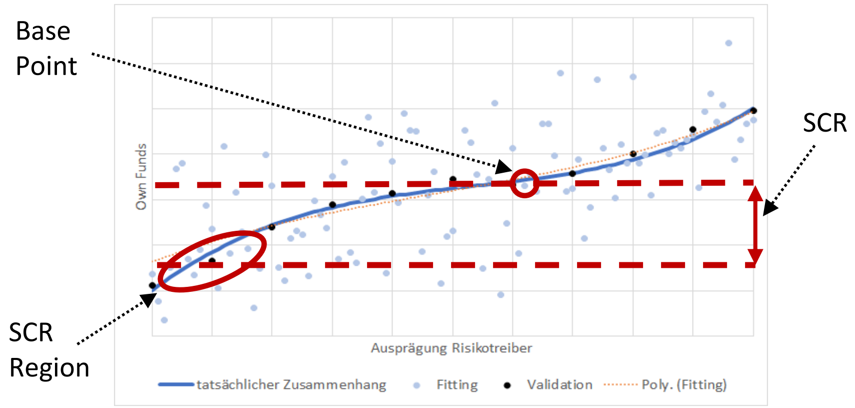


2. Data for Validation



256 one-year scenarios with **1,000** simulations

3. Base points and SCR Region



- Base point with **16,000** simulations
- **129** (portfolios 1 and 2) or **50** (portfolio 3) one-year scenarios around the 0.5% percentile scenario ("SCR region"), with **4,000** simulations

Jupyter Notebook

GitHub: Code and Data

3. Polynomials as Proxy Functions

LSMC with Polynomials

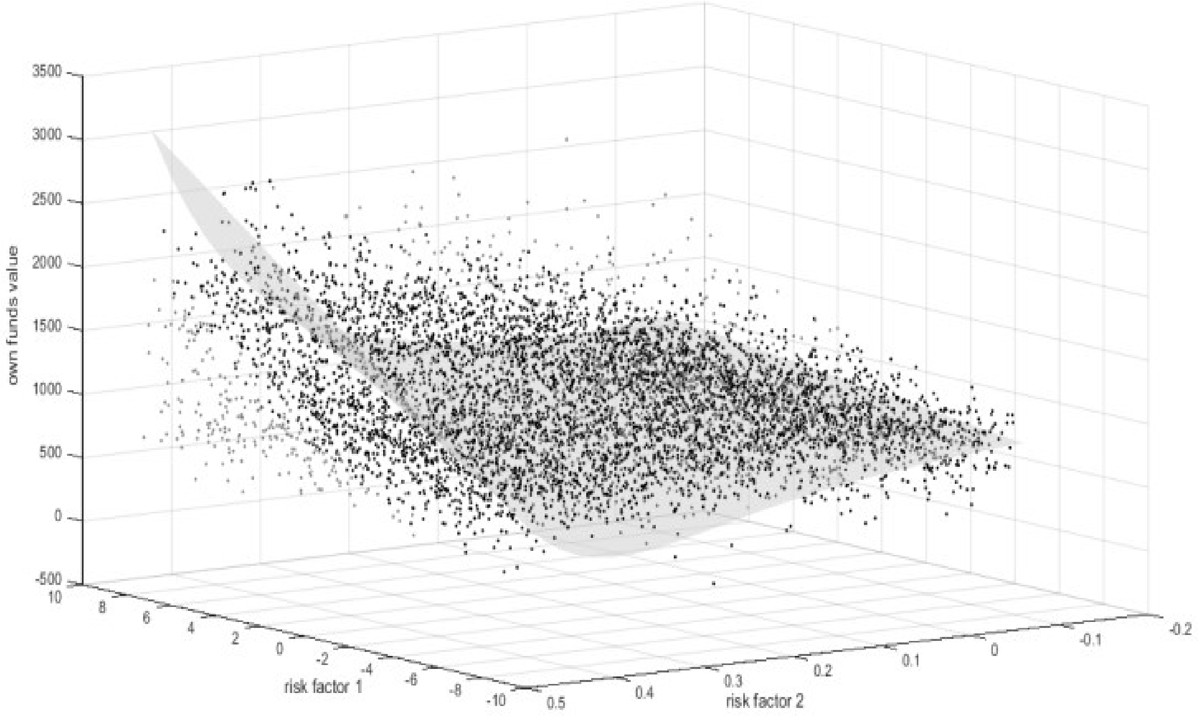
Current state of proxy modeling in the European market

A **model function** is given by a simple linear combination of the basis functions $\{\varphi_m(\cdot)\}_{m=1}^{M}$ with coefficients a_m:

$$f(RF_1, \dots, RF_D) = \sum_{m=1}^{M} a_m \varphi_m(RF_1, \dots, RF_D)$$

Let $PVFP_n,\ n = 1, \dots, N$ stand for the estimated values for Own Funds, and let $RF_{1n}, \dots, RF_{Dn}$ denote the N simulated risk factor vectors (one-year scenarios), then the function f can be determined using **least squares**:

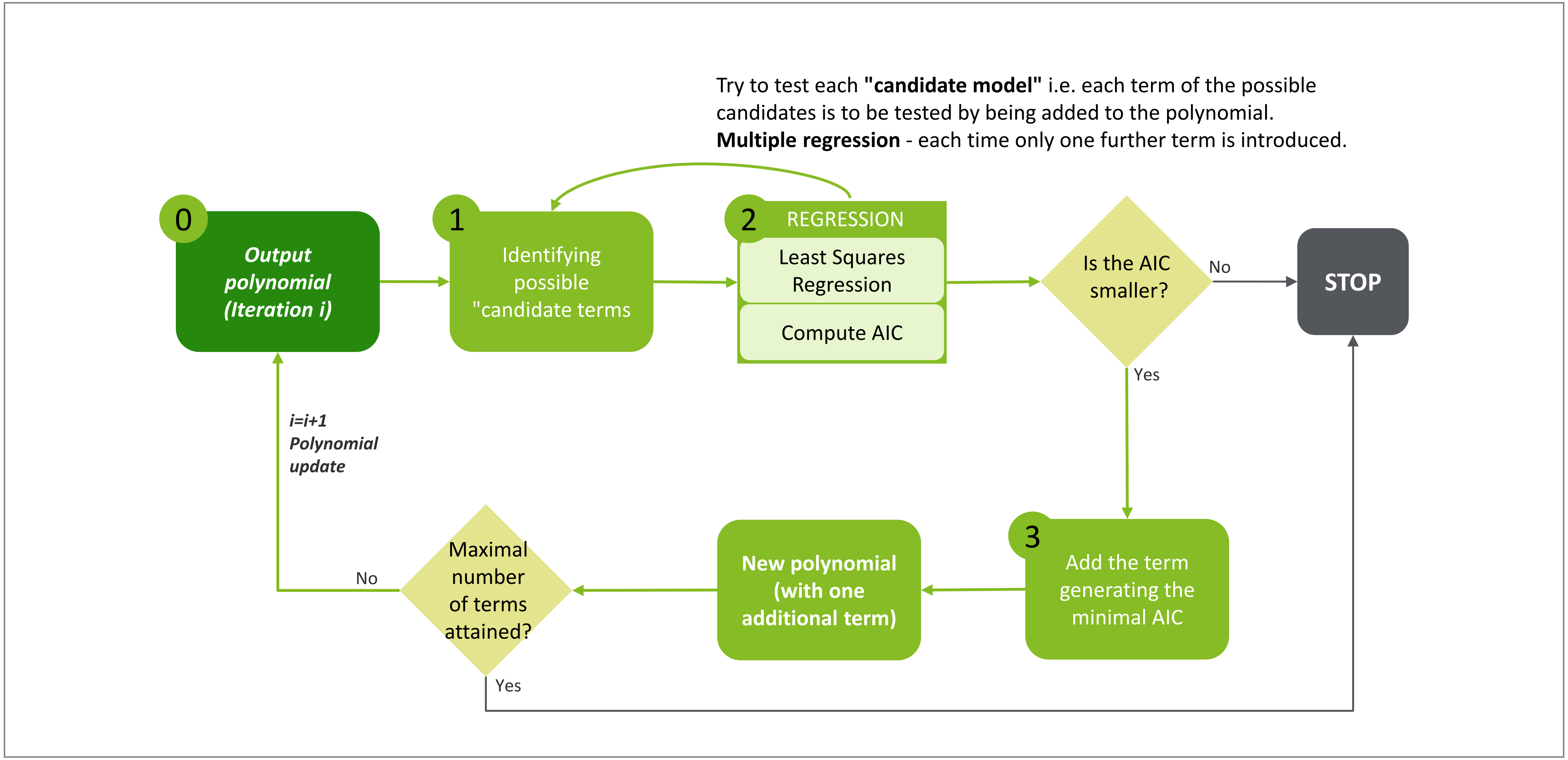
$$\min_{a \in R^M} \frac{1}{N} \sum_{n=1}^{N} (PVFP_n - f(RF_{1n}, \dots, RF_{Dn}))^2$$



 Source: Longstaff Schwartz, 2001

Model Selection Process

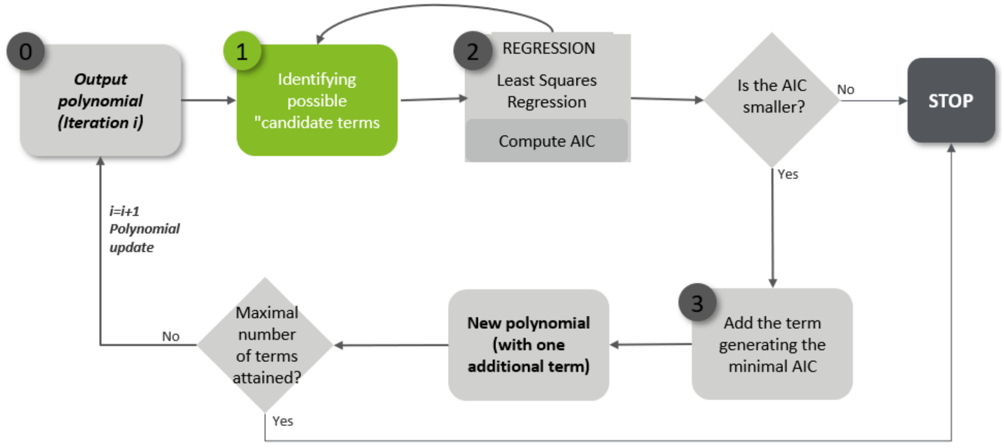
The Stepwise Algorithm



Example: Selection of Candidate Terms

The Stepwise Algorithm

Reminder:



Principle of Marginality

"A candidate term can be added to the model ***only if*** *the respective model* ***already contains all terms that are marginal to the candidate****."*

Choice of candidate terms using the adaptive multilevel method

A term may be added to the model only if the model already contains all factors of the term.

Example: model with three risk factors z_1, z_2 and z_3. At a certain step in the model selection process, the model consists of:

$$y = \beta_{0,0,0} + \beta_{1,0,0} z_1 + \beta_{0,1,0} z_2 + \beta_{2,0,0} z_1^2$$

The marginal terms that could be added to the model are then:

$$z_3, z_1 z_2, z_2^2, z_1^3, z_2 z_1^2$$

For example, the term $z_1 z_3$ could only be added if z_3 is already in the model.

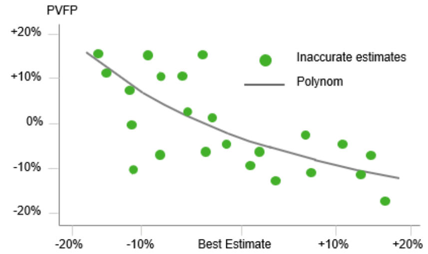
4. Neural Networks as Proxy Functions

Neural Networks

LSMC allows different regression functions

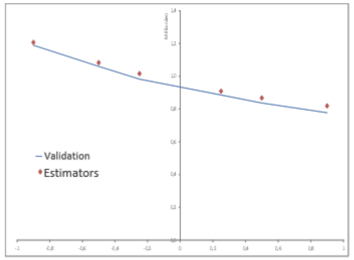
2c **Regression**

- Regression of a multidimensional function :
  - **Proxy-functions** for the PVFPs
- Risk drivers: e.g. costs in the image below



2d **Validation**

- Validation of the proxy-function
  - Via out-of-sample-tests
  - With 1.000 or 4.000 risk-neutral simulations



The polynomial functions in 2c and 2d are replaced with a neural network.

The underlying mathematical-finance and actuarial methodologies remain unchanged.

Neural Networks

The shortest possible mathematical introduction

Feed Forward Neural Network

Let $N_0, \dots, N_{L+1} \in \mathbb{N}$. A **fully connected feed forward neural network** with $L \in \mathbb{N}$ **hidden layers** is a function $f: \mathbb{R}^{N_0} \rightarrow \mathbb{R}^{N_{L+1}}$ defined as

$$f = (\Phi_{L+1} \circ a_{L+1}) \circ (\Phi_L \circ a_L) \circ \dots \circ (\Phi_1 \circ a_1),$$

where $\circ$ denotes the concatenation and $a_l: \mathbb{R}^{N_{l-1}} \rightarrow \mathbb{R}^{N_l}, l \in \{1, \dots, L+1\}$ are affine mappings represented by matrices W_l of dimension $N_{l-1} \times N_l$ and vectors $b_l \in \mathbb{R}^{N_l}$.

N_0 is the input dimension, N_{L+1} the output dimension and N_l the number of neurons or nodes in the hidden layer l.

Activation Functions

For each $l \in \{1, \dots, L+1\}$ the functions Φ_l are defined as

- $\Phi_l: \mathbb{R}^{N_l} \rightarrow \mathbb{R}^{N_l}$
- $\Phi_l(z_1, \dots, z_{N_l}) = (\phi(z_1), \dots, \phi(z_{N_l}))$,

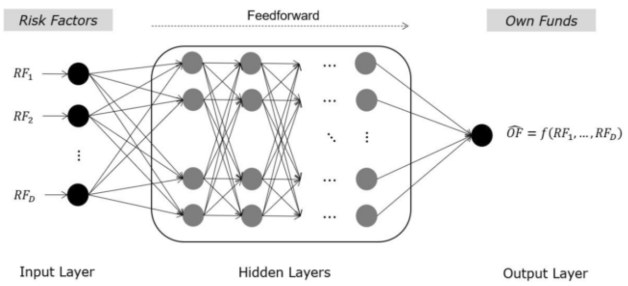
with real-valued functions $\phi_l: \mathbb{R} \rightarrow \mathbb{R}$, which are called **activation functions** of neural network and for which it is required to be monotone and Lipschitz continuous.

Examples:

- Sigmoid: $\phi(z) = \frac{1}{1+e^{-z}}$,
- Rectified linear unit (ReLU):
  $\phi(z) = \max(0, z)$,
- Leaky ReLU:
  $\phi(z) = \max(\lambda \cdot z, z)$
- Linear: $\phi(z) = z$

Training of Neural Networks

The functions a_l, $l \in \{1, \dots, L+1\}$ are affine transformations between the layers of the neural network and the parameters of the affine transformations are **weights** between the nodes of the layers. The change of these weights by an algorithm constitutes the **actual training** for given input and output data.



Neural Networks

Hyperparameter search leads to an ensemble

- **Hyperparameters:**
  - Settings provided to the algorithm that remain unchanged during training
  - They define the framework and structure of the model
- Examples: Learning rate, batch size, epochs, layers, activation functions, dropout rate, early stopping.
- Importance: Help optimizing model performance and preventing overfitting

- **Search Methods:**
  - Grid search over parameter grid
  - Random sampling of hyperparameters
  - Genetic algorithms and other

- **Example for Internal Model Data:**
- Train 108 neural networks with different hyperparameters
- Choose the best 10 models for the ensemble based on validation error
- The final value is the average of the values of 10 neural networks in the ensemble

- Effective hyperparameter tuning enhances model accuracy and robustness
- But it also increases the complexity of a model that anyway behaves like a black box

Neural Networks

Outcomes Compared

Method	Company 1	Company 2	Company 3
Polynomial only linear terms (Validation)	-17.8%	-0.4%	-0.1%
Polynomial only linear terms (SCR)	+12.0%	+0.9%	-3.7%
Quadratic terms also (Validation)	-6.8%	-0.5%	-0.1%
Quadratic terms also (SCR)	+27.4%	+1.6%	-2.3%
Terms selected by an expert (Validation)	-6.6%	-0.5%	-0.1%
Terms selected by an expert (SCR)	+20.8%	+0.5%	-1.7%
Stepwise algorithm with AIC (Validation)	-1.6%	-0.5%	+0.0%
Stepwise algorithm with AIC (SCR)	+6.7%	-0.9%	-1.3%
Ensemble of neural networks (Validation)	+0.2%	+0.1%	-0.0%
Ensemble of neural networks (SCR)	-1.1%	+0.4%	+1.0%

Conclusion

Conclusion

Data are available

- For complex actuarial projection models training and test data exist
- For three real portfolios data sets are available with a size that exceeds what most companies are able to produce

Proxy Modeling

- An explicit derivation of the Own Funds distribution over a one-year period close to impossible (“nested stochastics”)
- Financial mathematics and data science give us meaningful tools for approximation of the “heavy” actuarial model

Neural Networks Beat Polynomials

- The adaptive stepwise algorithm with polynomials is already advanced machine learning
- Ensembles of neural networks outperform the polynomials in this concrete task

Polynomials state-of-the-art proxy functions

- Successful implementation under the EU Solvency II legislation
- Usually faster and more reliable than other internal model proxy approaches

Questions