Calculations Guide

Atoti Market Risk

5.3

Table of Contents

Calculations Guide

This section provides calculation walk-throughs for Atoti Market Risk.

Component Measures

Contributory measures are used by Risk Managers to analyze the impact of a Sub-Portfolio on the Value at Risk of the total Portfolio. These measures can help to track down individual trades that have significant effects on VaR. Furthermore, contributory measures can be a useful tool in hypothetical analyses of portfolio development versus VaR development.

Basic Usage

In the below screenshot you can see a pivot table, displaying VaR and VaR Component BookHierarchy. The total of the Component VaR for the sub portfolios under the “Global Markets” node is equal to the VaR value computed for the “Global Markets”.

Level 1 / Level 2 / Level 3 Level 4 Level 5 Level 6 Level 7 Level 8 Level 9 Level 10 Level 11 Level 12	VaR Component BookHierarchy	VaR
(All) - Level 1		-593,128.88
Global Markets	-593,128.88	-593,128.88
Equities	-16,162.54	-70,520.70
FICC	-482,344.13	-610,621.82
Global Hedging	-94,622.20	-221,595.56

Sum: -593,128.88

Component VaR (CoVaR)

Definition

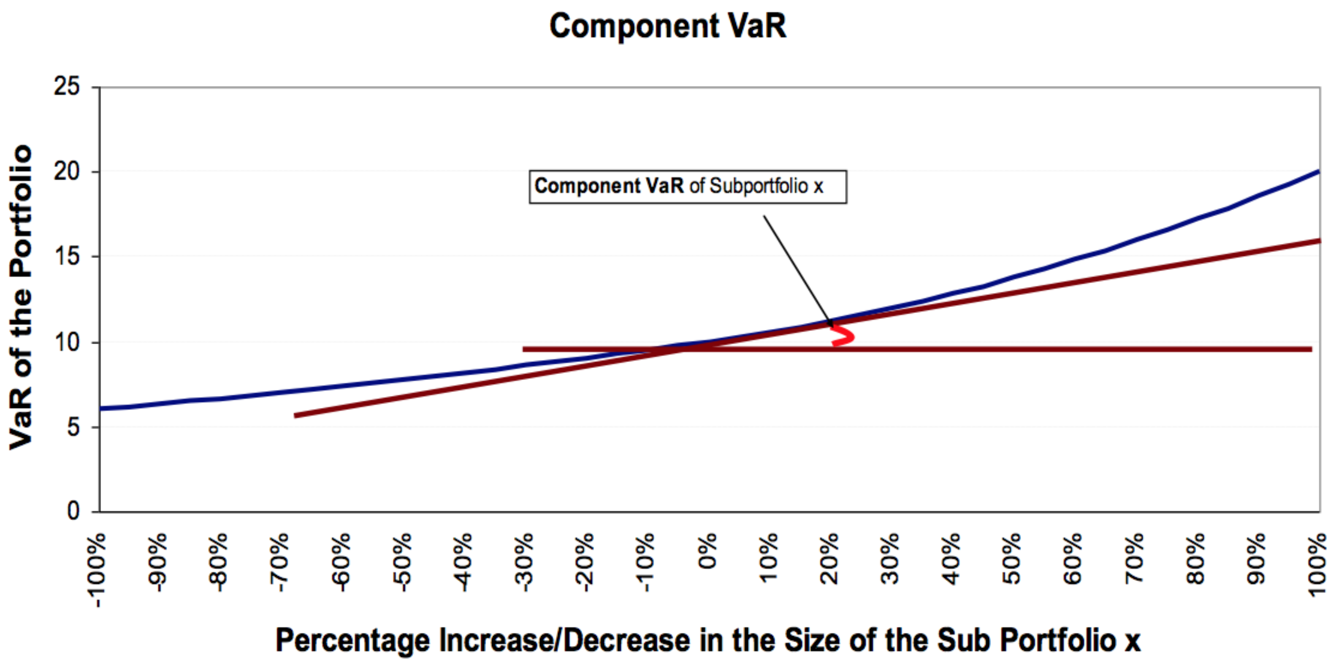
The Component VaR of Sub-Portfolio x measures the rate of change of the VaR of Portfolio X with respect to an incremental change in the size of Sub-Portfolio x.

$$CoVaR_x = \frac{\partial VaR(X)}{\partial w_x}$$

where w_x is the weight of Sub- Portfolio x in Portfolio X.

Component VaR is a very useful tool to examine the overall impact of local changes in a Sub-Portfolio upon the total Portfolio. It is an additive measure so that the Component VaRs of all Sub-Portfolios add up to the VaR of the total Portfolio.

In the Figure below, the CoVaR equals to the slope of the tangent line at point $(VaR_{wx}, w_x + 0)$.



Calculation Method

For a parent portfolio X with N components $Y_1, Y_2, ..., Y_N$ the algorithm[1] to compute CoVaRs is:

Regress:

$$Y = a + b \cdot X + c \cdot X^2$$

Calculate:

$$A = \begin{pmatrix} L & \sum x & \sum x^2 \\ \sum x & \sum x^2 & \sum x^3 \\ \sum x^2 & \sum x^3 & \sum x^4 \end{pmatrix} \rightarrow A^{-1}$$

A^{-1} - getting the inverse matrix.

Start Loop: For i = 1 to N:

1. Calculate:

$$B_i = \begin{pmatrix} \sum y_i \\ \sum x y_i \\ \sum y_i x^2 \end{pmatrix}$$

2. Multiply:

$$A^{-1} \cdot B_i = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

to get *a, b, c* the parameters of the regression.

3. Compute:

$$CoVaR_i = -\left[a + b \cdot (-VaR) + c \cdot VaR^2\right]$$

4. Compute:

$$CoVaR_i^{percentage} = \frac{-\left[a + b \cdot (-VaR) + c \cdot VaR^2\right]}{VaR}$$

End Loop: Next i

All sums, within matrices A and B, span for “ranked” vector scenario ids starting from 1 up to L, with 1 the most negative PnL value for the parent portfolio.

For example:

$$\sum_j x^2 = x_1^2 + x_2^2 + \ldots + x_L^2$$

$$\sum_j xy_i = x_1 y_{i1} + x_2 y_{i2} + \ldots + x_L y_{iL}$$

where

$$x_j$$

is the jth PnL of the “ranked” parent portfolio scenario ids and

$$y_{ij}$$

is its ith component’s PnL value on jth scenario id.

L denotes the number of scenario ids included in the regression.

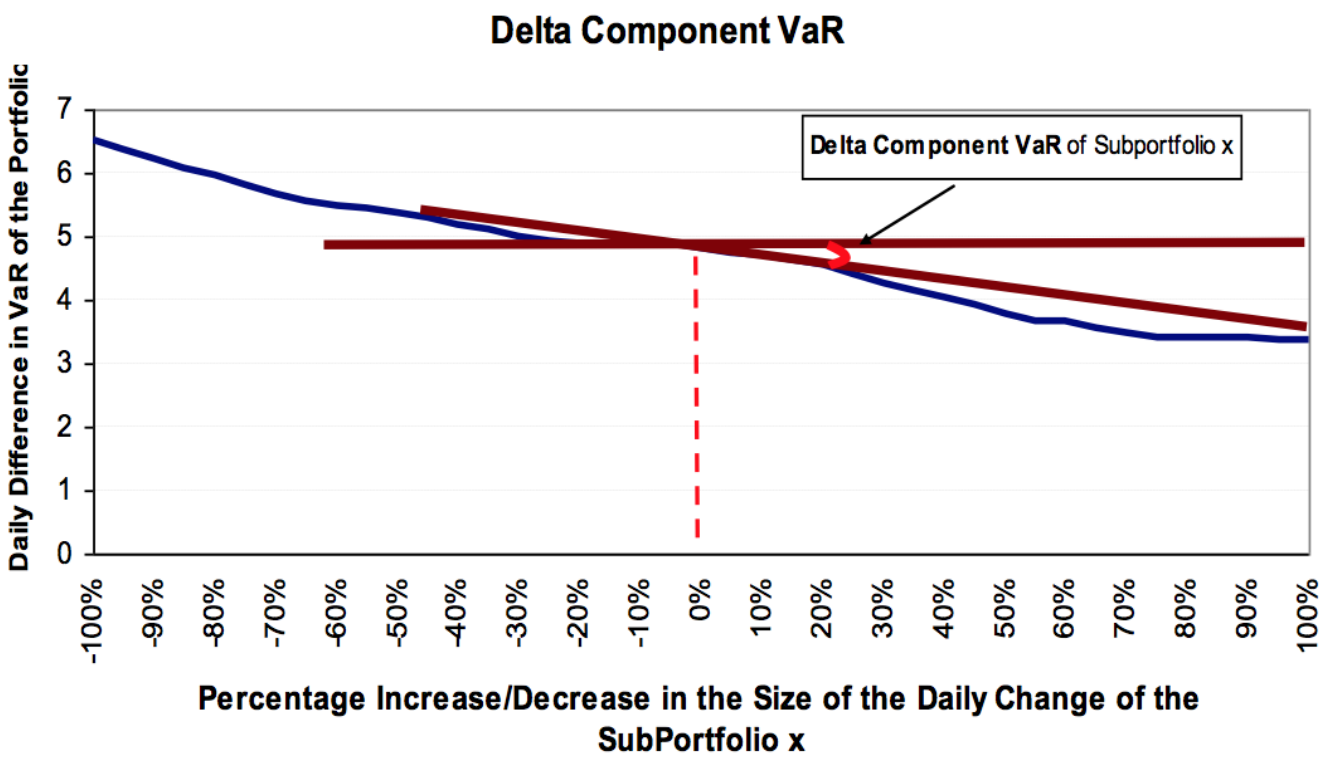
Delta Component VaR (Delta CoVaR)

The Delta Component VaR of Sub-Portfolio x measures the rate of change of a Portfolio’s daily change in VaR (Delta VaR(X)) with respect to an incremental change in a Sub-Portfolio’s daily change in size (Delta wx).

The Delta Component VaR is a very useful tool to examine the overall impact of local changes in a Sub-Portfolio upon the total Portfolio’s daily changes in VaR. Delta Component VaR is an additive measure so that the Delta Component VaRs of all Sub Portfolios add up to the daily change in VaR of the total Portfolio.

$$DeltaCoVaR_x = \frac{\partial(\Delta VaR(X))}{\partial(\Delta w_x)}$$

In the Figure below, CoVaR equals to the slope of the tangent line at point (ΔVaR, Δwx+0% * Δwx).



Calculation Method

Delta Component VaR or Delta CoVaR will be computed in the same fashion as the CoVaR by using the quadratic regression technique.

For a parent portfolio X with N components $Y_1, Y_2, ..., Y_N$ the algorithm to compute DeltaCoVaRs is:

Calculate:

$$A = \begin{pmatrix} L & \sum\Delta x & \sum\Delta x^2 \\ \sum\Delta x & \sum\Delta x^2 & \sum\Delta x^3 \\ \sum\Delta x^2 & \sum\Delta x^3 & \sum\Delta x^4 \end{pmatrix} \rightarrow A^{-1}$$

A^{-1} - getting the inverse matrix.

Start Loop: For i = 1 to N;

1. Calculate:

$$B_i = \begin{pmatrix} \sum\Delta y_i \\ \sum\Delta x\Delta y_i \\ \sum\Delta y_i\Delta x^2 \end{pmatrix}$$

2. Multiply:

$$A^{-1} \cdot B_i = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

to get *a, b, c* the parameters of the regression.

3. Compute:

$$\Delta CoVaR_i = -\left[a + b \cdot (-\Delta VaR) + c \cdot \Delta VaR^2\right]$$

4. Compute:

$$\Delta CoVaR_i^{percentage} = \frac{-\left[a + b \cdot (-\Delta VaR) + c \cdot \Delta VaR^2\right]}{\Delta VaR}$$

End Loop: Next i

The PnL vector inputs for the parent portfolio X with N components $Y_1, Y_2, ..., Y_N$ are now equal to the daily PnL changes from one business date to the next business date.

The changes of the parent portfolio are related to the changes to its N components with the following equations:

$$x_i^{cob} - x_i^{cob-1} = \left(y_1 i^{cob} - y_1 i^{cob-1}\right) + \left(y_2 i^{cob} - y_2 i^{cob-1}\right) + \ldots + \left(y_N i^{cob} - y_N i^{cob-1}\right), \text{ for i=1 to 500}$$

where

$$x_i$$

is the most backdated PnL date at cob for parent portfolio X.

All sums, within matrices A and B, span for "ranked" vector scenario ids starting from 1 up to L, with 1 the most negative PnL value for the parent portfolio.

For example,

$$\sum_j \Delta x^2 = \Delta x_1^2 + \Delta x_2^2 + \ldots + \Delta x_L^2$$

and

$$\sum_j \Delta x \Delta y_i = \Delta x_1 \Delta y_{i1} + \Delta x_2 \Delta y_{i2} + \ldots + \Delta x_L \Delta y_{iL}$$

, where

$$\Delta x_j$$

is the daily change of the jth PnL of the "ranked" parent portfolio scenario ids and

$$\Delta y_{ij}$$

its ith component's PnL value on jth scenario id.

ΔVaR is the change in VaR of the parent node from cob to cob-1.

L denotes the number of scenario ids included in the regression.

Algebraic solution

Here is the algebraic solution for the a, b, c.

$$a = \frac{\sum y \sum x^2 - \sum xy \sum x + c \sum x^3 \sum x - c\left(\sum x^2\right)^2}{n \sum x^2 - \left(\sum x\right)^2}$$

$$b = \frac{n \sum xy - nc \sum x^3 + c \sum x^2 \sum x - \sum y \sum x}{n \sum x^2 - \left(\sum x\right)^2}$$

$$c = \frac{\left(\sum x^2 y\right)\left(n \sum x^2 - \left(\sum x\right)^2\right) - \left(\sum x^2\right)^2 \sum y + \sum xy \sum x \sum x^2}{2 \sum x^3 \sum x \sum x^2 - \left(\sum x^2\right)^3 - n\left(\sum x^3\right)^2 + \sum x^4 \left(n \sum x^2 - \left(\sum x\right)^2\right)} + \frac{-n \sum x^3 \sum xy + \sum y \sum x \sum x^3}{2 \sum x^3 \sum x \sum x^2 - \left(\sum x^2\right)^3 - n\left(\sum x^3\right)^2 + \sum x^4 \left(n \sum x^2 - \left(\sum x\right)^2\right)}$$

1. The regression parameters a, b, c can also be computed algebraically. See Algebraic solution.

Corporate Actions

The corporate actions that occur on the underlying risk factors are taken into account by Atoti Market Risk. There are two main types of actions that can also be extended:

- Cash type: drop of a dividend or detachment of a coupon.
- Quantity type: split or merge of the instrument.

Input file

The corporate actions are stored in a specific file, as they occur infrequently. See CorporateAction.csv

Datastore

The corporate actions are also stored in a specific table. See CorporateAction table

Retrieval

See the Corporate Action section on the Market Data Retrieval Service.

PnL for Cash / Dividends

The dividends must be taken as cash contributing to the PnL for the current day.

Configuration

Element	Details

Element	Details
Input file column	'CashDividend'
Datastore column	'CashDividend'
Applied formula	Described under the properties of type `mr.sensi.rules.dividend`. The formula is applied with the following parameters for the current day: `cash=f(sensi, dividend, 0.0)`
Dividend	Described as a measure, similarly to the other greeks. Its configuration can be found on the class `DividendGreekSensiCubeMeasureConfig`.

PnL for Split

The split of a stock will divide its price by the split ratio. So to calculate the PnL Explain correctly, the market price must be adjusted to the same scale as the previous market price.

Configuration

Element	Details
Input file column	'SplitRatio'
Datastore column	'SplitRatio'
Applied formula	Described in the function `InterpolationConfiguration#corporateActionOperator(context, sensitivityKind, sensitivityName, riskClass)`. The formula is applied with the following parameters for the current day : `newMarketPrice=f(marketPrice, ICorporateAction)`

Extending corporate actions

The interface `ICorporateAction` can be extended to add extra actions.

Cross sensitivity

For Vanna, the cross sensitivities are handled by applying the shift formula of both risk factors:

$$\text{PnL Explain} = f_{\text{marketData2}}(f_{\text{marketData1}}(sens, q_1, q_2), p_1, p_2)$$

Where q_n and p_n are the market data quotes for a given day n for risk factors q and p respectively.

Currently, cross-bucket sensitivity is not supported.

FX calculation theory

The article presents the theory on how to calculate the PNL of a deal that is in position between two foreign currencies.

Sample deal

Let's take a deal with the following attributes at t0 :

Deal	Currency	RiskClass	RiskFactor	Value
Deal1	CC1	MtM [1]		1000
Deal1	CC2	FX	Delta CC2	600

[1] **MtM** means Mark to Market, it is the current price of the product on the market. It means that the pricing of the product under market conditions will produce the MtM.

We have a deal that is evaluated at 1000 CC1 and with a sensitivity to CC2 of 600 CC1.

Market values

FX rates	t0	t1
CC0/CC1	1.2	1.25
CC0/CC2	10	9.8
CC2/CC1	0.12	0.127551

To calculate the 3rd currency exchange rate:

$$\mathtt{CC2/CC1} = \frac{\mathtt{CC0/CC1}}{\mathtt{CC0/CC2}}$$

Here is the exchange rate conversion, the exchange rate name is the unit. So, multiplying n CC1 by CC0/CC1 gives CC0.

```
1 CC1 = 1.2 CC0
```

Conversion to Cash equivalent

To price this product, we will use a proxy that has the same characteristics on the market. In a perfect market this proxy must have the same price as the input product. We will use a basket of two currencies : CC1 and CC2.

Fx	MtM	Delta	Total
CC1	1000	-600	400
CC2		72	72

The cash from delta are calculated as follows:

$$\mathrm{cash}_{delta,\mathrm{cc1}} = -delta_{\mathrm{cc1}}$$
$$\mathrm{cash}_{delta,\mathrm{cc2}} = delta_{\mathrm{cc1}} \cdot \mathrm{cc2}/\mathrm{cc1}$$

Let’s check that it is correct at t0

For the MtM

Currency	Cash	Cash in CC0	MtM
CC1	400	480	1000
CC2	72	720	
Total in CC0		1200	1200

For the Delta

Currency	Cash	Cash in CC1	Cash in CC1 with CC2 + 1%	Delta
CC1	400	400	400	
CC2	72	600	606	
Total in CC1		1000	1006	6 / 1% = 600

PNL explain calculation

Let's calculate the PNL explain from the cash equivalent at t1:

Currency	Cash	Cash in CC0	Cash in CC1	Cash in CC2
CC1	400	500	400	51.02
CC2	72	705.6	564.48	72
Total = MtM at t1		1205.6	964.48	123.02
MtM at t0		*1200*	*1000*	*120*
PNL		5.6	-35.52	3.02
Variation of MtM		0.47%	-3.55%	2.52%

VaR calculation

NOTE

Simply multiplying the PNL vector by the Spot will not produce the right PNL vector in the domestic currency.

$$\overrightarrow{PLN_{\mathrm{CC0}}} \neq \overrightarrow{PLN_{\mathrm{CC1}}} \cdot \mathrm{fx}_{\mathrm{CCO/CC1}}$$

The currency rate is a stochastic variable that has to be taken into account in the VaR calculation.

For each scenario of the PNL vector, a specific exchange rate has to be used. This exchange rate must be consistent with the corresponding scenario.

This will give us :

$$PNL_{\mathrm{CC0}}(i) = PNL_{\mathrm{CC1}}(i) \cdot \left(shift_{\mathrm{CC0/CC1}}(i) + 1\right) \cdot \mathrm{fx}_{\mathrm{CCO/CC1}}$$

with the shift defined as

$$shift_{\mathrm{CC0/CC1}}(day) = \frac{\mathrm{fx}_{\mathrm{CCO/CC1}}(day)}{\mathrm{fx}_{\mathrm{CCO/CC1}}(day-1)} - 1$$

VaR calculation and risk class

If the Risk Class axis is selected we must split the VaR between the FX risk class and the underlying risk class.

So the VaR will be split as follows :

$$\begin{cases} PNL_{\mathrm{CC0}}^{FX}(i) = PNL_{\mathrm{CC1}}^{Other}(i) \cdot shift_{\mathrm{CC0/CC1}}(i) \cdot \mathrm{fx}_{\mathrm{CCO/CC1}} \\ PNL_{\mathrm{CC0}}^{Other}(i) = PNL_{\mathrm{CC1}}^{Other}(i) \cdot \mathrm{fx}_{\mathrm{CCO/CC1}} \end{cases}$$

Related information

You can see the formula in Excel: FxCalculation.xlsx

FX Effect on VaR

Overview

Simply multiplying the PNL vector by the Spot does not produce the correct PNL vector in domestic currency.

$$\overrightarrow{PNL_{CC0}} \neq \overrightarrow{PNL_{CC1}} \cdot FX_{CC0/CC1}$$

The currency rate is a stochastic variable that has to be taken into account in the VaR computation. For each scenario of the PNL vector, you must use a specific exchange rate. This exchange rate must be consistent with the corresponding scenario.

Additionally, to account for the FX risk associated with a trade in another currency, the current mark-to-market (MTM) value of the trade is multiplied by the same FX shift and added to the PNL for the trade. Where MTM is not provided for a trade, the FX risk is not accounted for.

This will give us:

$$PLN_{CC0^{(i)}} = (PLN_{CC1^{(i)}} \cdot (shift_{CC0/CC1^{(i)}} + 1) + MTM_{CC1} \cdot shift_{CC0/CC1^{(i)}}) \cdot FX_{CC0/CC1}$$

with the shift defined as

$$shift_{CC0/CC1^{(day)}} = \frac{FX_{CC0/CC1^{(day)}}}{FX_{CC0/CC1^{(day-1)}}} - 1$$

VaR computation and risk class

If the Risk Class axis is selected, we must split the VaR between the FX risk class and the underlying risk class.

So the VaR is split like this:

$$\begin{cases} PNL_{CCO}^{FX}(i) = PNL_{CC1}^{Other}(i) \cdot shift_{CC0/CC1}(i) \cdot FX_{CC0/CC1} \\ PNL_{CCO}^{Other}(i) = PNL_{CC1}^{Other}(i) \cdot FX_{CC0/CC1} \end{cases}$$

The name of the FX risk class is given by the parameter `mr.fx.risk-class-member=FX`

The FX exchange produces an effect on VaR only if the RiskFactorId column is set in the [FxRates] datastore, otherwise only the spot exchange rate conversion is applied.

FX Rates Service

With the DisplayCurrency analysis hierarchy you can change the dashboard's currency for the Value-at-Risk, sensitivities, PL, and other measures.

> **NOTE**
>
> Some of the measures will remain unaffected when the reference currency is changed, the measures in this list display values in the native currency by design. Please refer to the individual measures documentation for more information.

Default Reference Currency

If the DisplayCurrency analysis hierarchy is not explicitly present in the MDX query, the results will be converted into the default calculation currency configured in the application properties (`mr.fx.common-currency`).

Market data

The FX Rates are defined along the [AsOfDate] dimension to allow for historical rate conversions.

Multiple data sets - official end-of-date, trader's marks, etc - can be provided, and each data set will have to list the applicable currency pairs. We don't assume that all rates will be against the same currency. Please refer to the FXRate Lookup section below for more information.

We expect that the rates are displayed in their natural way: the market convention for some currency pairs is to always be referenced in a certain way, such as always showing a rate that multiplies or divides one currency by the other. To simplify the coding, we have chosen a single unique way across all currency pairs such as always multiply: GBP/USD 1.55 means that the amount in GBP is multiplied by 1.55 to produce the amount in USD. Please refer to the [FX Conversion Formula] section below.

FX Conversion Formula

The conversion is applied according to this formula:

$$Value_{\text{reference currency}} = Value_{\text{native currency}} \cdot FXRate$$

FXRate Lookup

The $FXRate$ for converting the native currency value into the reference currency value is obtained based on the data in the FX Rates data store.

1. In most cases, the algorithm will simply lookup the rate based on these key fields: AsOfDate, BaseCcy, CounterCcy. Initially the algorithm will search for the rate that will have AsOfDate, NativeCurrency, ReferenceCurrency in the key fields. See the Direct Lookup example below.
2. If the rate was not found, the algorithm will try the indirect lookup - search rate by AsOfDate, ReferenceCurrency, NativeCurrency and take the reciprocal of the rate if found.
3. If the rate is still not found at this stage, the algorithm will compute the rate using the FX crosses via the "CommonCcy" configured in the application properties (`mr.fx.common-currency`):
4. Search for the rate from CommonCcy to NativeCurrency, let this result be referred to as "baseCcyComponent"
5. Search for the rate from commonCcy to the referenceCurrency, let this result be referred to as "counterCcyComponent"
6. Compute the FXRate as counterCcyComponent divided by baseCcyComponent. See the FX Crosses example below.

Direct Lookup Example

- Let's imagine you want to see the measures expressed in CHF - you select "CHF" using the DisplayCurrency analysis hierarchy.
- The delivered risk for a position is 100 EUR: 100 is the risk value in units of native value currency EUR.
- The delivered FxRates for the business date:

AsOfDate	BaseCcy	CounterCcy	FXRate
2019-01-01	EUR	CHF	1.0794

The risk in CHF will be displayed as 107.94 = 100 x 1.0794.

FX Crosses Example

- Let's imagine a user wishes to see the measures expressed in CHF - she selects "CHF" using the DisplayCurrency analysis hierarchy.
- The delivered risk for a position is 100 KZT: 100 is the risk value in units of native value currency KZT.
- The common currency is set to EUR, and the FX crosses will use EUR as the common currency.
- The relevant FxRates for the business date:

AsOfDate	BaseCcy	CounterCcy	FXRate

AsOfDate	BaseCcy	CounterCcy	FXRate
2019-01-01	EUR	CHF	1.0794
2019-01-01	EUR	KZT	370.0427

With CHF as the reference currency, the rate applicable to KZT exposure is computed as follows:

- baseCcyComponent = EUR/KZT = 370.0427
- counterCcyComponent = EUR/CHF = 1.0794

The KZT/CHF rate is computed as 1.0794/370.0427 = 0.002916961.

Hence, the risk in CHF will be displayed as 0.2916961 = 100 x 0.002916961.

Incremental Measures

The Incremental measures evaluate the impact of a trade or a group of trades ('current scope') on the grand total result, by comparing it with a computation as if the given trade or aggregate of trades were hypothetically removed.

$$M^{incremental}(\text{scope}) = M(\text{portfolio}) - M(\text{portfolio excl scope})$$

Example

In the following example, the VaR Incremental BookHierarchy measure shows the impact of the three sub-portfolios on the firm-level VaR number.

VaR Incremental BookHierarchy

Level 1 / Level 2 / Level 3 Level 4 Level 5 Level 6 Level 7 Level 8	VaR Incremental BookHierarchy
Global Markets	-593,128.88
Equities	15,414.57
FICC	-363,364.42
Global Hedging	72,940.30

Sum: 15,414.57

VaR when the "Equities" is filtered out

Level 1 / Level 2 / Le...	VaR
(All) - Level 1	-608,543.45
Global Markets	-608,543.45
FICC	-610,621.82
Global Hedging	-221,595.56

Sum: -608,543.45

- The left pivot table shows that the VaR Incremental BookHierarchy for the business line "Equities" is +15.414, which means, that this portfolio has a **positive** +15.414 impact on the firm-level VaR
- To validate the incremental measure for the business line "Equities", let's compute firm-level VaR with a filter excluding this node (on the right pivot table), then the VaR is -609k which is 15.414 lower (-593k - (-609k))

LEstimated Measures

The LEstimated VaR is a contributory measure. It is an additive measure such that the LEstimated VaRs of all Sub-Portfolios add up to the VaR of the parent Portfolio.

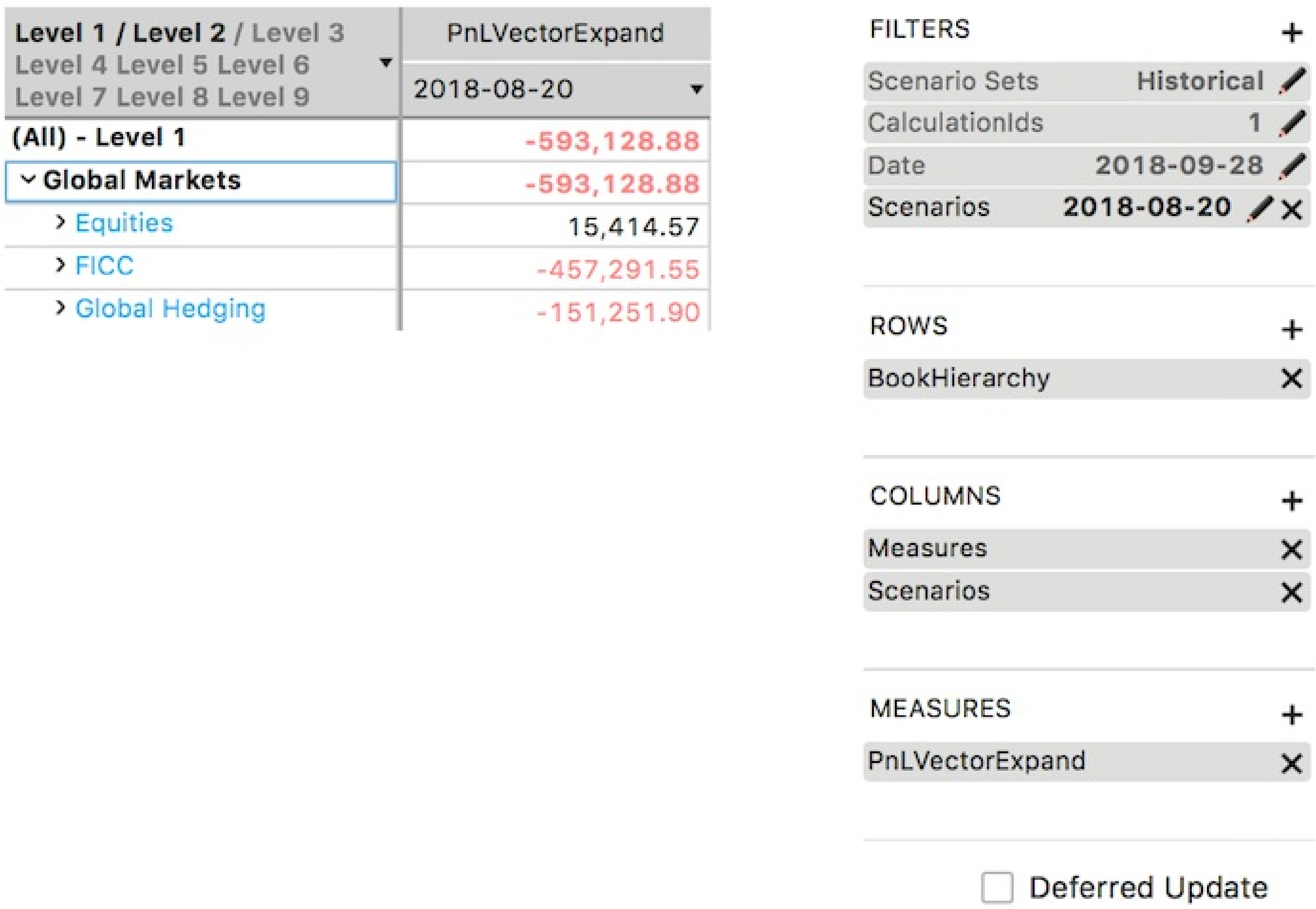
The LEstimated VaR shows the simulated PL for the tail scenario, that has been identified as the VaR scenario for the parent Portfolio.

In the screenshot below, a pivot table displays VaR and LEstimated VaR. The total of the LEsimated VaRs for the sub-portfolios under the "Global Markets" node is `-593k` and equals the VaR value computed for the "Global Markets". Although the VaR Scenario name(s) measure indicates that the VaR scenarios for each sub-portfolio were different, the LEstimated VaR has been based on the total portfolio VaR scenario `2018-08-20`.

Level 1 / Level 2 / Level 3 Level 4 Level	VaR	LEstimated VaR	VaR Scenario Name(s)
(All) - Level 1	-593,128.88		2018-08-20
Global Markets	-593,128.88	-593,128.88	2018-08-20
Equities	-70,520.70	15,414.57	2018-09-05
FICC	-610,621.82	-457,291.55	2018-03-07
Global Hedging	-221,595.56	-151,251.90	2017-11-16

Sum: -593,128.88

The following screenshot displays the PnLVectorExpand measure for the `2018-08-20` scenario. Indeed, the simulated PL values for the sub-portfolios match their LEstimated VaR values.



If the BookHierarchy is expanded further, the LEstimated VaR for the Level3 sub-portfolios will match the VaR Scenario computed for the Level2 parent, which is `2018-09-05` in our example.

Level 1 / Level 2 / Level 3 / Level 4	VaR	LEstimated VaR	VaR Scenario Name(s)
(All) - Level 1	-593,128.88		2018-08-20
Global Markets	-593,128.88	-593,128.88	2018-08-20
Equities	-70,520.70	15,414.57	2018-09-05
Cash Equities	-26,605.86	-21,271.18	2018-08-14
Volatility Trading	-57,726.97	-49,249.52	2018-08-17
FICC	-610,621.82	-457,291.55	2018-03-07
Global Hedging	-221,595.56	-151,251.90	2017-11-16

Level 1 / Level 2 / Level 3 / Level 4 Level 5 Level 6 Level 7 Level 8 Level 9 Level 10 Level 11 Level	PnLVectorExpand	
	2018-08-20	2018-09-05
(All) - Level 1	-593,128.88	51,074.84
Global Markets	-593,128.88	51,074.84
Equities	15,414.57	-70,520.70
Cash Equities	1,449.15	-21,271.18
Volatility Trading	13,965.42	-49,249.52
FICC	-457,291.55	130,047.26
Global Hedging	-151,251.90	-8,451.72

Sum: 15,414.57

Parametric VaR

The Parametric VaR calculation assumes that the PnL returns are normally distributed and also independent of each other. Consequently, the calculated standard deviation is used to compute a standard normal Z-score to determine the VaR.

Example of parametric VaR calculation:

- Standard deviation of PnL over specified time period: 25,000
- Mean of PnL over specified time period: 50,000
- Z-score for 99% confidence level: 2.326

The Parametric VaR for the specified time period with a 99% confidence level is:

50,000 - 25,000 * 2.326 = -$8,150

PnL Explain

The PnL Explain process aims to quantify the changes in the PnL from one business day to the next, based on the closing prices for each day and show the impact of each input (each change in the quote) in terms of PnL.

For example, the portion of PnL due solely to the change of one particular quote, such as the EUR SWAP LIBOR 3Y.

The process begins with aggregation of the greeks and then uses those values in a Taylor expansion to explain the expected one-day change in PnL using the actual market data shift over one day.

Input data

Atoti Market Risk supports Delta, Gamma, Vega, Volga, Vanna, Cash, and Theta sensitivities.

Market data

If the exact vertices/tenor points referenced by the sensitivities don't coincide with the market data, Atoti Market Risk can apply a linear interpolation of the market data quotes between sensitivities and market data tenors.

That interpolation logic can be enabled or disabled based on the risk class and the sensitivity measure. Also, additional computations can be defined for every combination of sensitivity measures and risk classes before and after the interpolation of market data is performed.

The market data must be at the risk-factor level. Atoti Market Risk performs any necessary interpolation and fallback.

Methodology

In Atoti Market Risk, the PnL Explain calculations are configurable and extendable. In the decomposition of PnL, the market data impact represents the change in the portfolio value from market data shifts during the day.

Shifts and shift factors

Atoti Market Risk supports both relative and absolute market data shifts. This is configurable and extendable. The shift factor default is 1. The following table shows the PnL impact calculations for each risk class, where q_n is the market data quote for a given day n.

Risk class	How sensitivities are represented	1st order Taylor approximation	2nd order Taylor approximation

Risk class	How sensitivities are represented	1st order Taylor approximation	2nd order Taylor approximation
Equity, FX	Cash equivalent positions	$(q_2/q_1 - 1) \cdot \text{sens}$	$\frac{1}{2}(q_2/q_1 - 1)^2 \cdot \text{sens}$
IR, credit, repo curves	Provided in $/bp	$(q_2 - q_1) \cdot \text{sens}$	$\frac{1}{2}(q_2 - q_1)^2 \cdot \text{sens}$
Vega	$/vol point	$(q_2 - q_1) \cdot \text{sens}$	$\frac{1}{2}(q_2 - q_1)^2 \cdot \text{sens}$

DHS computation

For negative interest rates, Atoti Market Risk uses the following DHS (Displaced Historical Simulation) computations, where a is a shift value that can be configured per risk class.

1st order Taylor approximation	2nd order Taylor approximation
$\text{sens} \cdot [(q_2 + a)/(q_1 + a) - 1]$	$([(q_2 + a)/(q_1 + a) - 1])^2 \cdot \text{sens}$

Cross sensitivity

For Vanna, the cross sensitivities are handled by applying the shift formula of both risk-factors:

$$\text{PnL Explain} = f_{\text{marketData2}}(f_{\text{marketData1}}(\text{sens}, q_1, q_2), p_1, p_2)$$

Where q_n and p_n are the market data quotes for a given day n for risk-factors q and p respectively.

> NOTE
>
> Currently, cross-bucket sensitivity is not supported.

Sensitivity ladders

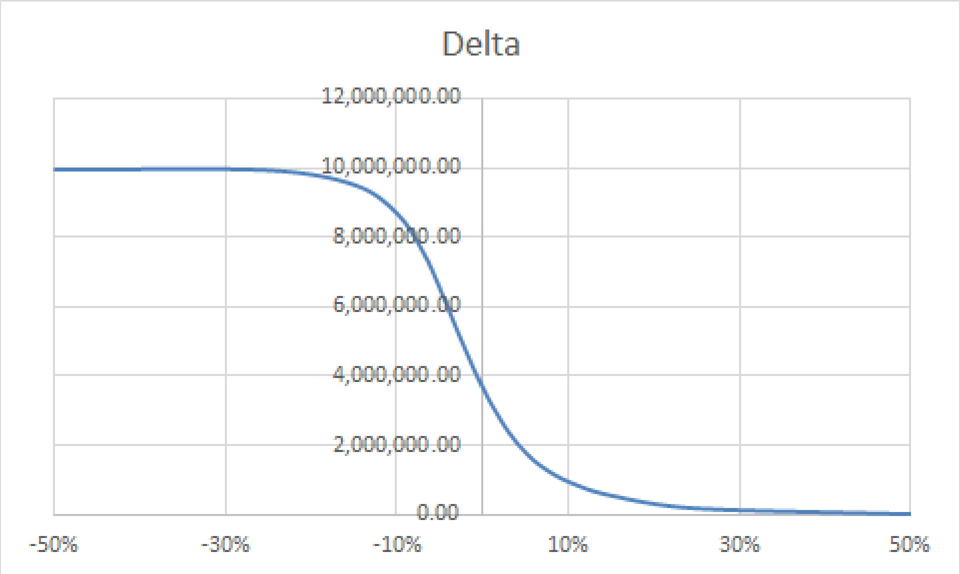
The purpose of the sensitivity ladder is to increase PnL calculation accuracy by computing the sensitivities at different market configurations. It is used to compute PnL Explain and the Taylor VaR.

Input data

The system is fed with a table of sensitivities shifted from the current market value.

For instance:

Shift	Delta
-50%	9,944,227.53
-25%	9,924,531.37
-15%	9,501,104.56
-10%	8,668,816.62
-7%	7,569,003.25
-5%	6,499,116.86
-3%	5,306,289.78
-1%	4,157,778.67
0%	3,633,050.69
1%	3,153,644.65
3%	2,353,529.88
5%	1,765,164.79
7%	1,348,405.85
10%	934,996.82
15%	539,326.70
25%	171,999.71
50%	2,613.78



Calculation theory

The purpose is to calculate the PnL based on the delta ladder for the underlying market price. The ladder is based on price shift, so for PnL explain we first convert the market data pair into a shift.

For PnL explain

The conversion between market data and shift is done with the following formula:

$shift_d = \text{ShiftFromMD}(MD_d, MD_{d-1})$

This formula is defined by the `IPnLExplainFormulaProvider::getShiftFromMDFormula` method bean, based on the *mr.sensi.rules.*.type* property.

Type	**ShiftFromMD formula**
Absolute	$shift_d = MD_d - MD_{d-1}$
Relative	$shift_d = \frac{MD_d}{MD_{d-1}} - 1$
FXRelative	$shift_d = \frac{MD_d}{MD_{d-1}} - 1$
DHS	$shift_d = \frac{MD_d + factorShift}{MD_{d-1} + factorShift} - 1$

WARNING

To keep consistency between ladders and standard sensitivities across PnL Explain and Taylor VaR, the formula provider bean must ensure that

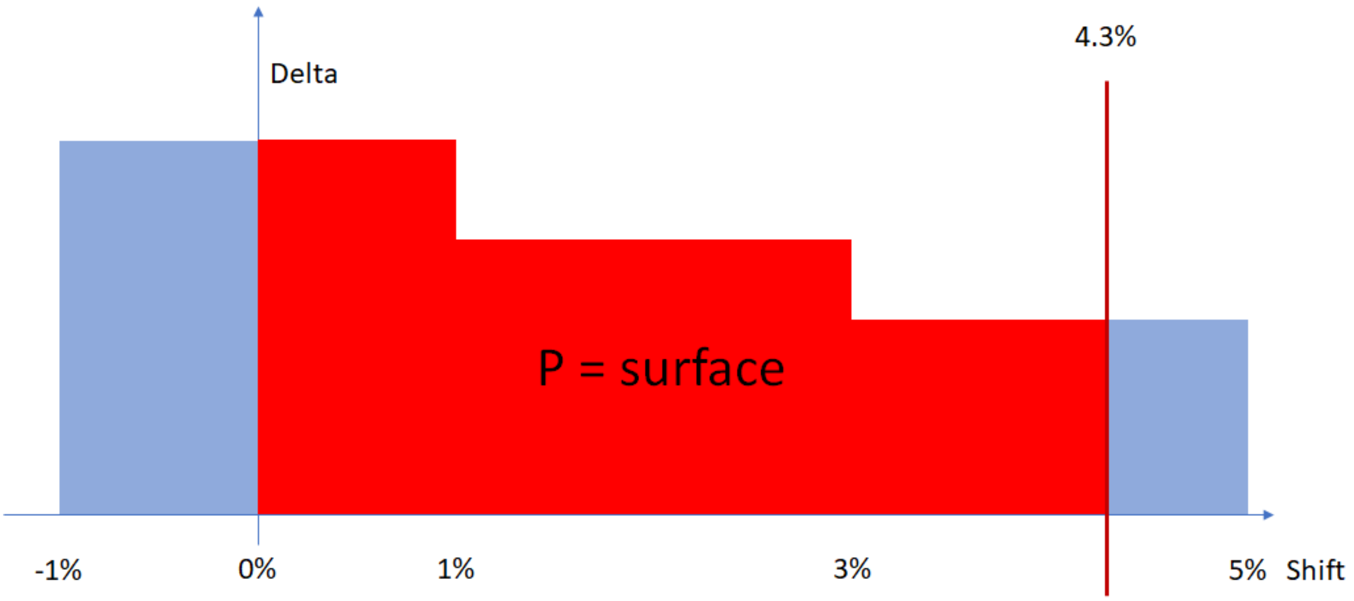
$\text{PnlExplain}(sensi, MD_d, MD_{d-1}) = \text{VaRExplain}(sensi, \text{ShiftFromMD}(MD_d, MD_{d-1}))$

- PnlExplain is provided by `IPnLExplainFormulaProvider::getPnlExplainFormula`
- VaRExplain is provided by `IPnLExplainFormulaProvider::getVaRExplainFormula`
- ShiftFromMD is provided by `IPnLExplainFormulaProvider::getShiftFromMDFormula`

Ladder computation

We assume that $P(shift) = P(0) + \int_0^{shift} delta(x) \cdot \partial x$. As we are on a discrete distribution (the ladder), this will be transformed into: $PnL(shift) = P(shift) - P(0) = \sum_{shift_i \in]0, shift]} \int_{shift_{i-1}}^{shift_i} delta(x) \cdot \partial x$

So it will lead to the integration of the Red surface:



Ladder integration parts

To compute the $\int_{shift_{i-1}}^{shift_i} delta(x) \cdot \partial x$ part of the formula, several methodologies can be used depending on the way the delta ladder has been computed. The Ladder module will call the partial integration computation for each ladder step. By default, Atoti Market Risk provides the following ways to compute this partial integration:

Formula Name	Function	Behavior
DeltaShift	LadderFromDeltaShift::ladder	This function supposes that the ladders are $L(x) = \frac{(P(x) - P(0))}{x}$ so $PnL(x) = x \cdot L(x)$, the ladder is interpolated between the plots.

Formula Name	Function	Behavior
LinearIntegration	LadderLinearIntegration::ladder	This function is computing the surface between two ladders plots as a trapeze to perform an integration of $L(x) \cdot \partial x$.
DerivativeRelative	LadderFromDerivativeRelative::ladder	This function supposes that $L(x) = \frac{\partial P(x)}{\partial x}$ and use it as a constant sensitivity until the next step for integration.
DerivativeAbsolute	LadderFromDerivativeAbsolute::ladder	This function supposes that $L(x) = \frac{x.\partial P(x)}{\partial x}$ and use it as a constant sensitivity until the next step for integration.

Adding a new formula

To add a new formula, create a new Extended Plugin that complies with the signature:

```
@QuartetExtendedPluginValue(intf = ILadderFormulaProvider.class, key = MyLadderFormula.PLUGIN_KEY)
public class MyLadderFormula implements ILadderFormulaProvider {

    public static final String PLUGIN_KEY = "MyLadderFormula";

    @Override public ILadderFormula getLadderFormula() {
        return MyLadderFormula::ladder;
    }

    public static double ladder(double shift, double shift1, double ladder1, double shift2, double ladder2, DoubleBinaryOperator pnlFormula,
            FormulaStep currentWindow) {
        //
        // Here comes you formula
        //
    }

    @Override public String getType() {
        return PLUGIN_KEY;
    }
}
```

The formula has to comply with the `ILadderFormula` interface. The function will be called and the result summed for each ladder step until the target shift is reached.

Parameter	Type	Meaning
shift	double	The target shift we curently compute expressed as $shift = \frac{MD_d}{MD_{d-1}-1}$
shift1	double	The shift of the 1rst ladder plot (closest to the center).

Parameter	Type	Meaning
ladder1	double	The ladder value of the 1rst plot.
shift2	double	The shift of the second ladder plot.
ladder2	double	The ladder value of the 2nd plot.
pnlFormula	DoubleBinaryOperator	The function `(sensi, shift) -> PNL` coming from `IPnLExplainFormulaProvider::getVaRExplainFormula`
currentWindow	FormulaStep	This enum provides the position of the shift against the current ladder interval. - BELOW: shift is between 0 and shift1 - IN: shift is between shift1 and shift2 - ABOVE: shift is above shift2 (only called when the shift is out of the ladder).

Implementation

Input files

Ladder pillar definition

For details on this input file format, see Ladder Definition.

Datastore

The ladder definition is stored as is in a standalone store.

The ladder sensitivities are stored in the TRADE_SENSITIVITIES datastore.

For a scalar model, each line is split by tenor, maturity, and moneyness according to the scalar model of data.

In the vectorized model, the lines are stored with the bucket expansion.

Cube

The Ladder column is populated by native measures in the Sensitivity Cube. The Cube also has an analysis dimension on the ladder definition, so with an Expand post-processor you can view the content of the ladder.

Calculation logic

The logic is customized with the properties under **mr.sensi.rules**. For details on these properties, see Sensitivities module properties.

The `InputSelector` bean is managing the way the Ladders are applied.

The function `IInputSelector::getFormulaInput(@NonNull LadderContext context, @NonNull String sensitivityKind, @NonNull String sensitivityName, @NonNull String riskClass)` defines whether the Ladder methodology is applied following the `SensitivityInput` value by reading the `ladder` properties:

Ladder	Calculation
SENSI_ONLY	The formula is applied exclusively to sensitivity inputs, excluding ladders.
LADDER_FIRST	The formula attempts to use ladders as input, defaulting to sensitivities if ladders are not available.
LADDER_ONLY	The formula is applied exclusively to ladder inputs.

The function `IInputSelector::getLadderFormula(@NonNull String sensitivityKind, @NonNull String sensitivityName, @NonNull String riskClass)` will return the formula used for ladders depending on the selected methodology defined on the `ladder-formula` properties.

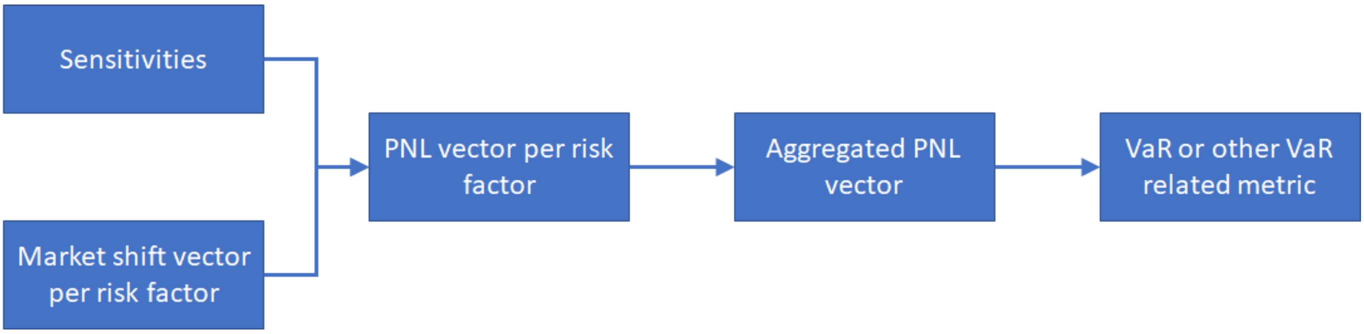
Taylor VaR

Overview

With the sensitivities it is possible to compute an approximated PnL of a product using the Taylor formula. $f(a+h) = f(a) + \nabla f(a) \cdot h + \frac{1}{2} h^T \mathbb{H}(a) h + o(\|h\|^2)$ In this formula, the gradient vector is the first-order risk factor vector and the Hessian matrix is the second-order risk factor matrix.

Cube process

In the cube, the PnL computation with the Taylor formula is not performed per trade, but instead per risk factor, by applying the Taylor part of the formula to this specific risk factor / market shift pair. The applied formula is the same as the one used for the PnL Explain functionality. As the next stage of the formula is purely linear, with the help of cube aggregation, we can output PnL per trade, risk factor, and so on. The market shift is provided directly as a vector compatible with the VaR scenarios.



Sources

The risk factor values are read straight from the aggregated sensitivity metrics. The market shifts are loaded directly from the MarketShifts store.

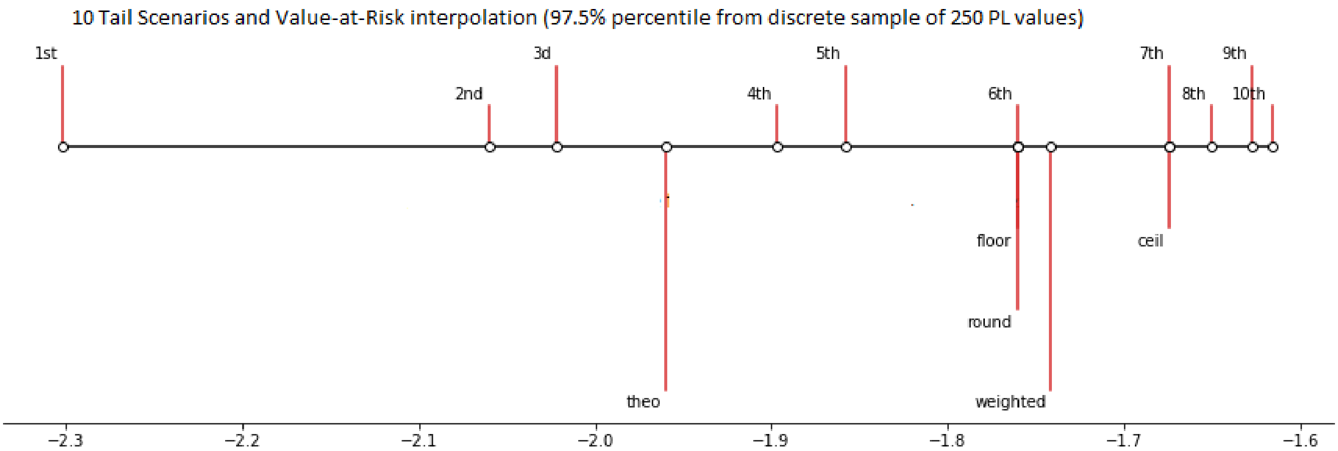
Limitations

The market shifts must be the same kind as the sensitivity, that is, either absolute or relative.

VaR Interpolation

The percentile is a well-defined concept in the continuous case, but there are multiple ways to compute it from a discrete sample.

Consider the following example. This is a plot of the 10 smallest values in a sample of 250 values drawn from the standard normal distribution. The plot shows the theoretical 2.5% quantile of the standard normal distribution ("theo") as well as various types of 97.5 percentile estimation (floor, ceil, weighted, etc).



Atoti Market Risk supports multiple types of VaR estimation, as described in this chapter.

Calculation steps

The algorithm can be described as follows:

- Compute quantile as (1-confidence level)
- Compute adjacent ranks based on Quantile2Rank setting, quantile and discrete sample size
- Sort PL and obtain PL values for the adjacent ranks approximate VaR from the PL values using the interpolation formula controlled by the `rounding.var` property

Computing adjacent ranks

The property “rounding.quantile2Rank” can be used to control the rank calculation:

Property value	Description	Formula for rank	Example	Notes
rounding.quantile2Rank =CENTERED	This corresponds to the “First variant, C=1/2” of the interpolation variants described on this wiki	x = quantile * vectorSize + 0.5	(1-97.5%) x 250 + 0.5 = 6.75	
rounding.quantile2Rank =EQUAL_WEIGHT	This corresponds to the “Third varian, C=0” of the interpolation variants described on this wiki	x = quantile * (vectorSize+1)	(1-97.5%)x (250+1) = 6.275	default option
rounding.quantile2Rank =EXCLUSIVE		x = quantile * (vectorSize+1)-1	(1-97.5%)x (250+1)-1 = 5.275	

Based on the computed x, we can obtain various adjacent ranks:

- x_lower = round_down(x)
- x_higher = round_up(x)
- x_nearest = round(x)
- x_nearest_even = round_to_even(x)
- weight = x mod 1

VaR approximation

The property `rounding.var` can be used to control the interpolation.

Property value	Description	Formula for VaR	Notes
rounding.var =FLOOR	Simulated PL for the lower of the adjacent ranks	VaR = PL_lower	

Property value	Description	Formula for VaR	Notes
rounding.var =CEIL	Simulated PL for the higher of the adjacent ranks	VaR = PL_higher	default option
rounding.var=WEIGHTED	Simulated PL linearly interpolated between the adjacent ranks	VaR = weight * PL_lower + (1-weight) * PL_higher	
rounding.var =ROUND	Simulated PL for the nearest rank VaR = PL_nearest		
rounding.var=ROUND_EVEN	Simulated PL for the nearest even rank	VaR = PL_nearest_even	

where PL values are defined as follows (in the sorted PL vector):

- PL_lower as x_lower smallest value
- PL_higher as x_higher smallest value
- PL_nearest as x_nearest smallest value
- PL_nearest_even as x_nearest_even smallest value

WHS

Atoti Market Risk provides the “Weighted” variations of the VaR and ES measures which implement the **WHS** - exponentially weighted historical simulation approach.

References

The implemented algorithm is described in Section 3 of the following paper:

- Richardson, Matthew P. and Boudoukh, Jacob and Whitelaw, Robert F., The Best of Both Worlds: A Hybrid Approach to Calculating Value at Risk (November 1997).

Implementation

1. The default value for the **decay parameter** λ is 0.94, it is configured in the `mr-config.properties` and can be overridden using the WeightedVaRLambda context value.
2. Each historical scenario is assigned a weight, computed based on the λ value and the number of elapsed business days.
3. The simulated PL input data, which can be displayed using the PnLVectorExpand measure, is ranked from the worst loss to the highest profit.
4. The scenario weights are accumulated starting from the worst loss and further along the scenarios

ranked by PL.

5. The Weighted VaR is obtained by linearly interpolating the PLs of the ranked scenarios where accumulated weights contain the desired VaRConfidenceLevel.
6. The Weighted ES is obtained by averaging the PL across scenarios below the Weighted VaR. Please note however, that the confidence level for the Weighted ES is controlled by a different context value - ESConfidenceLevel.