CCruncher - Technical Document

Gerard Torrent Gironella

Version 1.5 - R586

Abstract

The ccruncher goal is to compute the credit risk of a portfolio, where investments are fixed income assets, taking into account default correlations between economic sectors. This is done by determining the probability distribution of portfolio loss at time T using the Monte Carlo method and computing the usual risk statistics (Expected Loss, Standard Deviation, Value at Risk, Expected Shortfall).

Keywords: credit risk, Monte Carlo, survival functions, gaussian copula, t-Student copula, value at risk, expected shortfall, block matrix.

Contents

1 Parameters

This section explains the input parameters required by CCruncher.

1.1 Ratings and Survival Functions

A rating tells a lender or investor the probability of the subject being able to pay back a loan. A poor rating indicates a high risk of defaulting. Every rating has a survival function associated. This function indicates the probability that a borrower with initial rating X be non-defaulted at time t. The creation of a rating system and the construction of the survival functions are outside the scope of this paper.

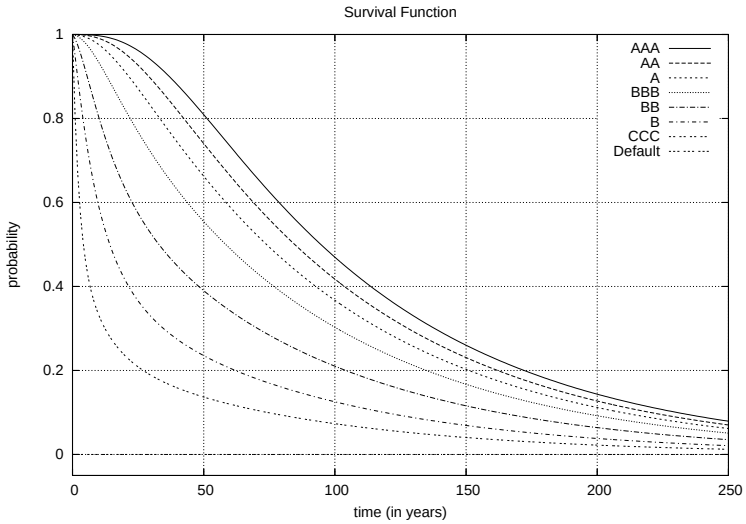

Figure 1: Survival functions

The survival data is often not available, and where it is available, it is often inconsistent or not representative. For this reason it is better to work with the transition matrix assuming that the rating transitions follow a Markov model. Appendix A.1 shows how to determine the survival functions using the transition matrix.

1.2 Sectors and Correlations

The risk of a credit portfolio depends crucially on default correlations between economic sectors. These sectors are groupings of companies that react similarly to given economic conditions. Example of sectors: energy, financial, technology, media and entertainment, utilities, health care, etc. Default correlations between

sectors measures the defaults times dependence between the defined sectors. This can be expressed in tabular form:

	$Sector_1$	$\dots$	$Sector_m$
$Sector_1$	$\rho_{1,1}$	$\dots$	$\rho_{1,m}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$
$Sector_m$	$\rho_{1,m}$	$\dots$	$\rho_{m,m}$

$$\rho_{i,j} = Corr(Sector_i, Sector_j)$$

Note that diagonal elements are default time correlations between members of the same sector (values distinct than 1) while elements outside diagonal are default time correlations between elements of distinct sectors.

Appendice A.3 expose a simple method to obtain an estimation of the correlation matrix based on the historical information about defaults.

1.3 Portfolio

Portfolio is composed by borrowers. Each borrower has an initial rating and belongs to a sector. Each borrower have one or more assets. Each asset is defined by its addition date in the portfolio, expected cashflow and recovery at certain dates. Figure 3 shows the structure of a portfolio.

Cashflow indicates the cash given to the borrower (negative amounts) and the cash received from the borrower (positive amounts) at each date (see figure 2).

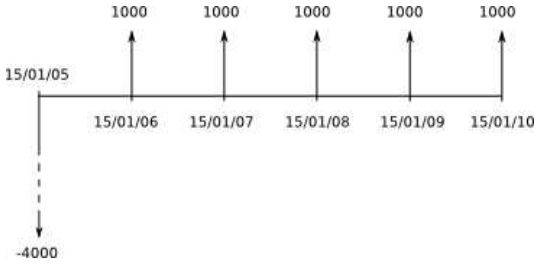

Figure 2: Asset cashflow example

Recovery is a ratio that indicates the portion of loss that will be recovered in case of default at a fixed time. This value takes into account guarantees, the cost to take legal actions, recovery rate based on historical data, etc.

If a borrower defaults, the loss is the sum of all remaining cashflows at default time (*exposure*) weighted by recovery rate at default time.

$$\text{loss}_t = (1 - \text{recovery}_t) \cdot \sum_{i \geq t}^{N} \text{cashflow}_i$$

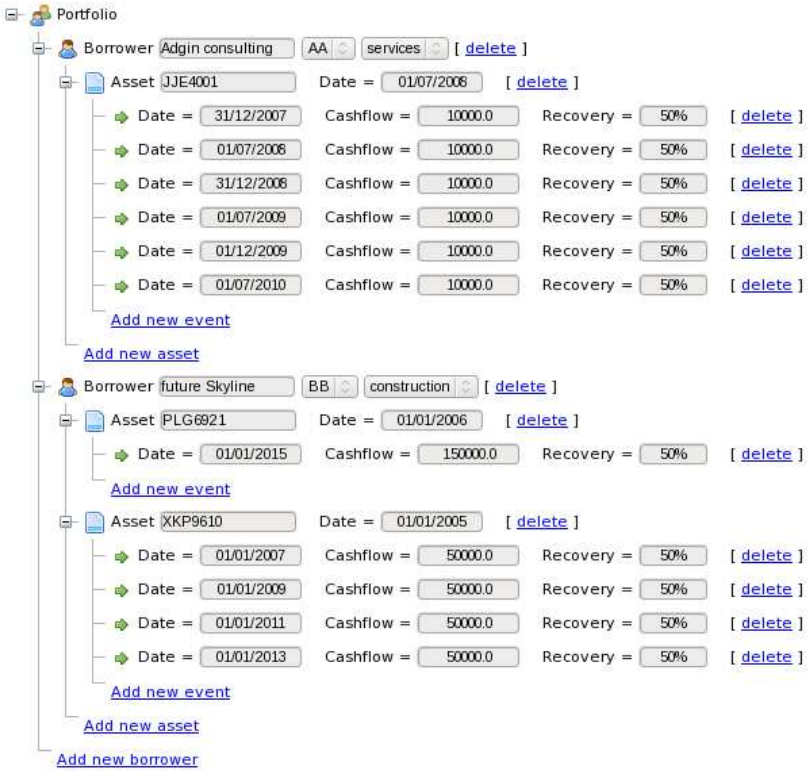

Figure 3: Portfolio example

2 Resolution

This section explains how CCruncher computes the credit risk of a portfolio.

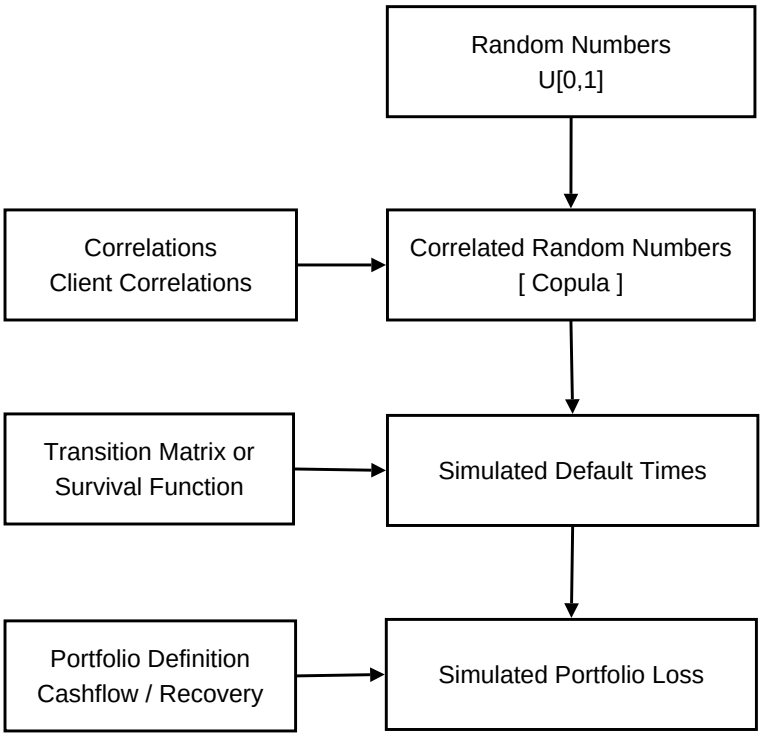

Figure 4: Monte Carlo simulation schema

2.1 Borrowers correlation matrix

We need to translate from default correlations between sectors to default correlations between borrowers. Let us suppose that we have n borrowers and m sectors. Each borrower belongs to a sector. Correlations between sectors are known (see section 1.2):

	$Sector_1$	$\dots$	$Sector_m$
$Sector_1$	$\rho_{1,1}$	$\dots$	$\rho_{1,m}$
$\vdots$	$\vdots$	$\ddots$	$\vdots$
$Sector_m$	$\rho_{1,m}$	$\dots$	$\rho_{m,m}$

$$\rho_{i,j} = Corr(Sector_i, Sector_j)$$

We sort the borrowers so that we find those of sector 1 at the beginning and those of sector m at the end. Then we create the borrowers correlation matrix taking

as correlation between two borrowers the correlation between their sectors (see figure 5).

$$\Sigma = \begin{pmatrix}
1 & \cdots & \rho_{1,1} & & \rho_{1,k} & \cdots & \rho_{1,k} & & \rho_{1,m} & \cdots & \rho_{1,m} \\
\vdots & \ddots & \vdots & & \vdots & & \vdots & & \vdots & & \vdots \\
\rho_{1,1} & \cdots & 1 & & \rho_{1,k} & \cdots & \rho_{1,k} & & \rho_{1,m} & \cdots & \rho_{1,m} \\
 & & & \ddots & & & & & & & \\
\rho_{1,k} & \cdots & \rho_{1,k} & & 1 & \cdots & \rho_{k,k} & & \rho_{k,m} & \cdots & \rho_{k,m} \\
\vdots & \ddots & \vdots & & \vdots & \ddots & \vdots & & \vdots & & \vdots \\
\rho_{1,k} & \cdots & \rho_{1,} & & \rho_{k,k} & \cdots & 1 & & \rho_{k,m} & \cdots & \rho_{k,m} \\
 & & & & & & & \ddots & & & \\
\rho_{1,m} & \cdots & \rho_{1,m} & & \rho_{k,m} & \cdots & \rho_{k,m} & & 1 & \cdots & \rho_{m,m} \\
\vdots & \ddots & \vdots & & \vdots & \ddots & \vdots & & \vdots & \ddots & \vdots \\
\rho_{1,m} & \cdots & \rho_{1,m} & & \rho_{k,m} & \cdots & \rho_{k,m} & & \rho_{m,m} & \cdots & 1
\end{pmatrix}$$

Figure 5: Borrowers correlation matrix

Observe that this is a correlation matrix (symmetric, $|\rho_{i,j}| \leq 1$, $|\rho_{i,i}| = 1$) composed by blocks. This matrix will be decomposed using the Cholesky algorithm, so it must be definite positive.

2.2 Monte Carlo simulation

Monte Carlo methods are a class of computational algorithms for simulating the behavior of various physical and mathematical systems. Each simulation consists of computing a random default time for each borrower and then calculate the loss of the portfolio. We need that default times fulfill two conditions: the borrower's survival functions and the borrower's correlation matrix. This is achieved by simulating a copula.

2.2.1 Random numbers generation

A copula [12] [14] is a multivariate random variable where each component is a uniform $U[0,1]$. Each simulation requires a set of random numbers between $[0,1]$ and correlated by the borrowers' correlation matrix, Σ (see figure 5).

$$\begin{array}{|c|c|c|c|} u_{11} & u_{12} & \dots & u_{1n} \\ u_{21} & u_{22} & \dots & u_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ u_{k1} & u_{k2} & \dots & u_{kn} \end{array}$$

n number of borrowers
k number of simulations
$u_{.i} \sim U[0,1]$
$Corr(u_{.i}, u_{.j}) = \Sigma_{ij}$

There are implemented two copula generators, the gaussian and the t-Student copulas. Apendices A.4 and A.5 describes the algorithms used to generate them.

Exists evidences [4] in favor of the t-Student copula. In comparison with the t-Student copula, gaussian copula underestimates the probability of joint extreme downward movements. Moreover gaussian copula is too optimistic on diversication benefits.

2.2.2 Default times simulation

Given the borrower k and the random number $u_k \in [0,1]$ (generated using a copula in the previous step) we simulate the default time considering the inverse of the borrower survival function at u_k [3]. Figure 6 shows this in a graphic manner.

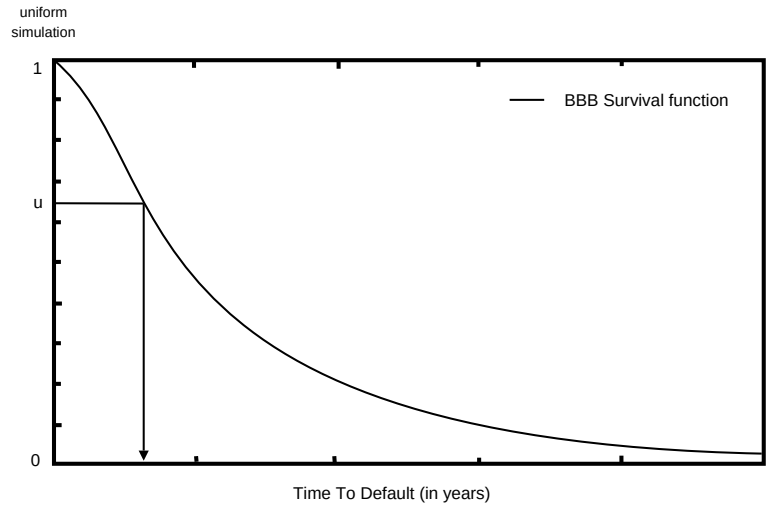

Figure 6: Default time generation with initial rating BBB

2.2.3 Portfolio loss evaluation

At this point, given any borrower, we have a simulated default time. The loss caused by this borrower is the sum of all remaining cashflows at default time weighted by recovery at default default time. To obtain the simulated value of the portfolio loss we sum all borrower losses.

$$\text{PortfolioLoss} = \sum_{i=1}^{N} \text{BorrowerLoss}_i$$

Where N is the number of borrowers in the portfolio. For each of them we use the simulated default time, t_i, described in the previous section to compute the borrower loss.

$$\text{BorrowerLoss}_i = (1 - \text{recovery}_{t_i}^i) \cdot \sum_{j \geq t_i}^{E^i} \text{cashflow}_j^i$$

2.3 Risk computation

After N simulations (eg. 20000, 500000 or more) we have a list of numbers, $x_1, ..., x_N$, where each number represents a simulated portfolio loss. The more values, the more accuracy in the results. All risk statistics (eg. Expected Loss) have an error margin that will be estimated. CCruncher uses *R package*[1] to perform the statistical computations described below.

[1] http://www.r-project.org

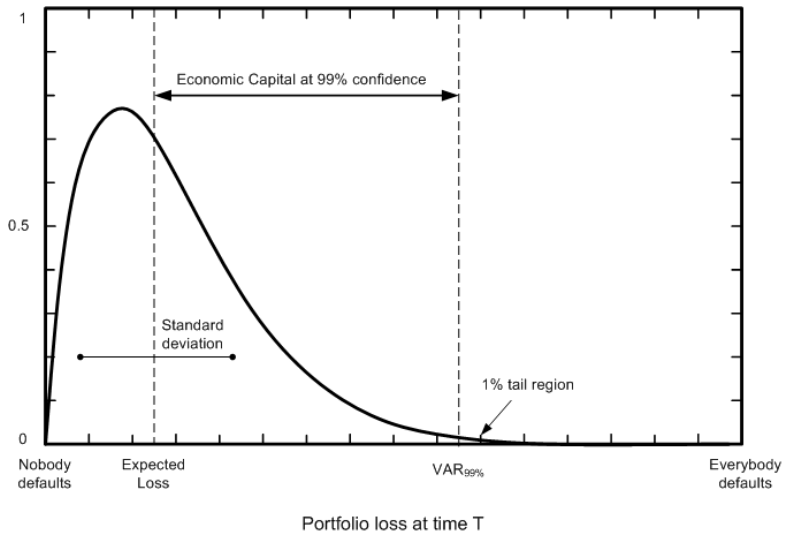

Figure 7: Portfolio loss at time T

2.3.1 Expected Loss

Expected Loss is mean of the portfolio loss distribution. Central Limit Theorem [13] grants that:

$$\mu = \widehat{\mu} \pm \phi^{-1}\left(\frac{1-\alpha}{2}\right) \cdot \frac{\widehat{\sigma}}{\sqrt{N}}$$

where α is the error confidence level, ϕ^{-1} the N(0,1) inverse cumulative distribution function and $\widehat{\mu}$ and $\widehat{\sigma}$ are the mean and stddev estimators:

$$\widehat{\mu} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\widehat{\sigma} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}\left(x_i - \widehat{\mu}\right)^2}$$

2.3.2 Portfolio Loss Standard Deviation

Another usual risk statistic is the standard deviation of the portfolio loss. Central Limit Theorem [13] grants that:

$$\sigma = \widehat{\sigma} \pm \phi^{-1}\left(\frac{1-\alpha}{2}\right) \cdot \frac{\widehat{\sigma}}{\sqrt{2N}}$$

where α is the error confidence level, ϕ^{-1} the N(0,1) inverse cumulative distribution function and $\widehat{\sigma}$ is the stddev estimator defined previously.

2.3.3 Value At Risk

Value at Risk [9] is the most used risk value. We call it VAR_β where β is the VAR confidence level (eg. VAR at 95%). VAR is another form to say quantile. Then $VAR_\beta = q_\beta = \inf\{x|F(x) \geq \beta\}$.

$$VAR_\beta = \widehat{q_\beta} \pm \phi^{-1}\left(\frac{1-\alpha}{2}\right) \cdot \text{stderr}(q_\beta)$$

where α is the error confidence level, β is the VAR confidence level, ϕ^{-1} the N(0,1) inverse cumulative distribution function, $\widehat{q_\beta}$ is the quantile estimator, and stderr(q_β) is the estimation of the standard error.

$$\widehat{q_\beta} = x_{k:N}$$

where,

- k fulfills $\frac{k}{N} \leq \beta < \frac{k+1}{N}$
- $x_{k:N}$ is the k-th element of ascendent sorted values

We determine stderr(q_β) using Maritz-Jarret method described in [7].

$$\begin{aligned} M &= [N\beta + 0.5] \\ a &= M - 1 \\ b &= N - M \\ W_i &= B(a, b, \frac{i+1}{N}) - B(a, b, \frac{i}{N}) \\ C_k &= \sum_{i=1}^{N} W_i \cdot x_i^k \end{aligned}$$

where $[x]$ is the integer part of x and $B(a, b, x)$ is the incomplete beta function:

$$B(a, b, x) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \int_0^x t^{a-1}(1-t)^{b-1} dt$$

Then,

$$\text{stderr}(q_\beta) = \sqrt{C_2 - C_1^2}$$

2.3.4 Expected Shortfall

VAR is not a distance because it does not fulfill the sub-additive property [8], $VAR(A+B) \nleq VAR(A) + VAR(B)$. Expected Shortfall is a consistent risk measure [2] similar to VAR. It can be described as the average of the $\beta\%$ worst losses.

$$ES_\beta = \widehat{ES_\beta} \pm \phi^{-1}\left(\frac{1-\alpha}{2}\right) \cdot \text{stderr}(ES_\beta)$$

where α is the error confidence level, β is the ES confidence level, ϕ^{-1} the N(0,1) inverse cumulative distribution function, $\widehat{ES_\beta}$ is the ES estimator and stderr(ES_β) is the estimation of the standard error.

We select the simulation portfolio loss values $(x_1, ..., x_N)$ that are bigger than VAR_β.

$$y_1, y_2, y_3, \cdots, y_K \qquad \text{where} \quad y_i > VAR_\beta$$

Then,

$$\widehat{ES_\beta} = \frac{1}{K}\sum_{i=1}^{K} y_i$$

$$\text{stderr}(ES_\beta) = \frac{\sqrt{\frac{1}{K-1}\sum_{i=1}^{K}\left(y_i - \widehat{ES_\beta}\right)^2}}{\sqrt{K}}$$

2.3.5 Economic Capital

We can compute the Economic Capital at confidence level β as:

$$\text{Economic Capital} = VAR_\beta - \text{Expected Loss}$$

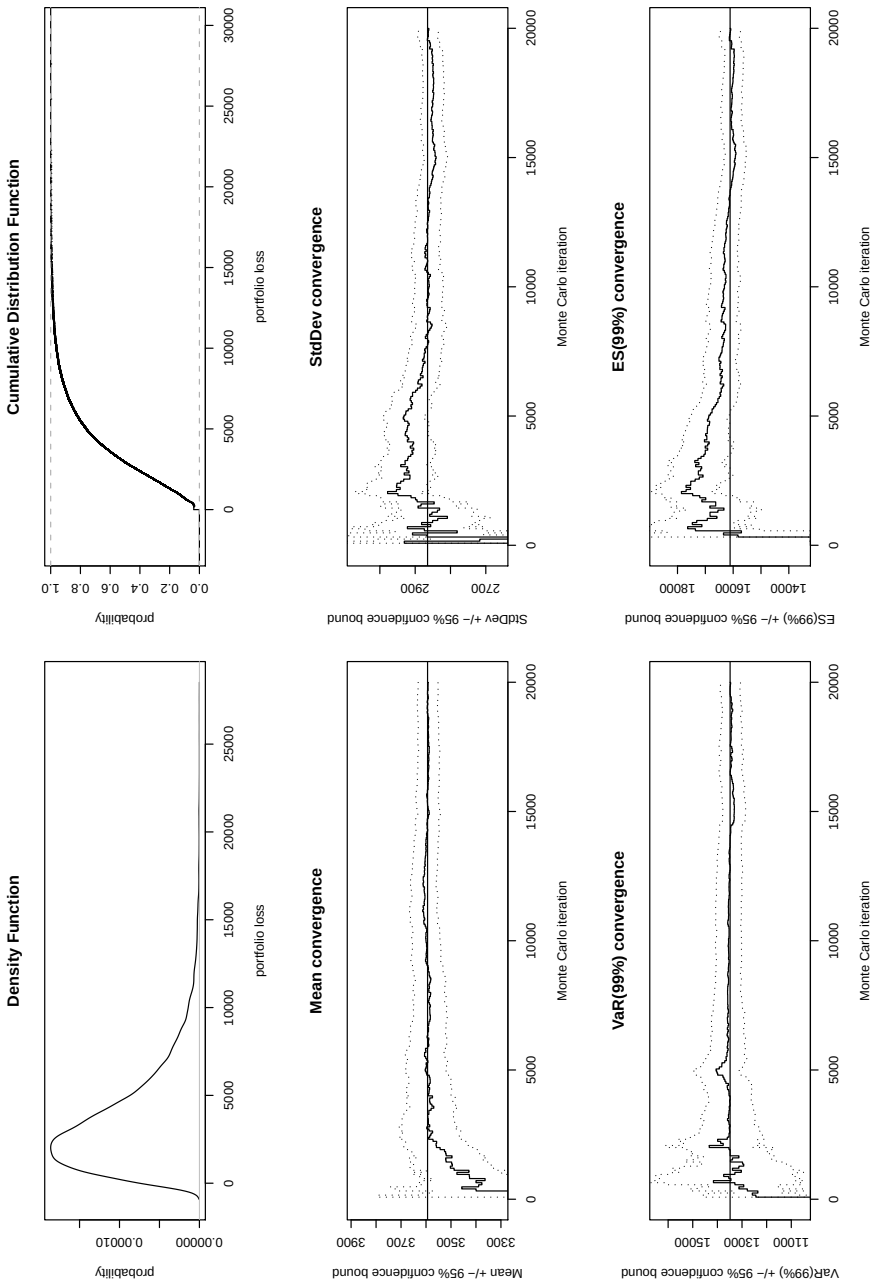

Figure 8: CCruncher results

3 Other considerations

CCruncher takes into consideration other concepts that have not been exposed up to this moment with the purpose of simplifying the content.

3.1 Risk aggregation

CCruncher simulates the whole portfolio loss at time T. It also allows (in the same execution) the simulation of subportfolios by defining one or more aggregators. Consequently you can determine which are the borrowers (or products or branches or regions or assets, etc.) that increase the risk of your portfolio. Every subportfolio has its own list of simulated values that can be processed to obtain the risk indicators for this subportfolio.

3.2 Yield curve

CCruncher can consider that money value decreases along the time following a fixed curve (eg. yield curve). CCruncher allows the input of the yield curve and and takes it into account in all its computations. The coeficient that determines the money value decreases along the time is given by the compound interest formula:

$$\Upsilon(r, t_0, t_1) = (1 + r)^{(t_1 - t_0)}$$

3.3 Antithetic technique

The random number generation using a copula is time expensive. Gaussian and t-Student copula are symmetric, that means that $(u_1, u_2, \cdots, u_n)$ is equiprobable to $(1 - u_1, 1 - u_2, \cdots, 1 - u_n)$. In CCruncher's antithetic mode, each copula generation is used 2 times, $(u_1, u_2, \cdots, u_n)$ and $(1 - u_1, 1 - u_2, \cdots, 1 - u_n)$, reducing to the half the number of generated copulas.

3.4 Parallel computing

Monte Carlo problems are embarrassingly parallel problems. In the jargon of parallel computing, an embarrassingly parallel workload (or embarrassingly parallel problem) is one for which no particular effort is needed to segment the problem into a very large number of parallel tasks, and there is no essential dependency (or communication) between those parallel tasks.

In CCruncher the master sends the RNG seed to each slave ($seed_i = seed + i \cdot 30001$) and waits for results (the simulated values of the portfolio loss) from slaves

that store them in a file. When one of the two stop criteria is achieved (maximum execution time exceded or maximum number of simulations) the master sends a stop signal to the slaves and leaves.

4 Numerical example

Lets go to compute the credit risk at $T = 30$ years for the following portfolio.

Id	Borrower name	Rating	Sector	Recovery	Assets
B1	Future Skyline	AA	Construction	80%	A1
B2	Ramses builds	BBB	Construction	75%	A2, A3
B3	Gramdateresi	AA	Consumer goods	60%	A4
B4	Tantor Pinori	BB	Consumer goods	90%	A5, A6, A7
B5	White & Rebolan	B	Services	30%	A8
B6	Adign Consulting	BBB	Services	60%	A9
B7	Advanced Engineering	AAA	Services	50%	A10
B8	Sigmacle Research	CCC	Services	60%	A11
B9	Loglament Asia	AA	Services	50%	A12

Table 1: Portfolio composition

The following table list the assets events in months from current date.

Month	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12
6	10$				100$							70$
12		50$				20$	80$		20$			
36											30$	
60		30$			10$		20$	80$				
90						30$						
120	15$		40$		30$							
180					40$	70$						
216		10$			50$							80$
240				100$								
312	100$		40$		70$				70$			
360						15$					40$	
384	90$				90$					100$	90$	

Table 2: Assets events

In this example we supose that we don't dispose of survival functions but we have the 1-year transition matrix T_1 (extracted from [6]).

	AAA	AA	A	BBB	BB	B	CCC	Default
AAA	90.81	8.33	0.68	0.06	0.12	0.00	0.00	0.00
AA	0.70	90.65	7.79	0.64	0.06	0.14	0.02	0.00
A	0.09	2.27	91.05	5.52	0.74	0.26	0.01	0.06
BBB	0.02	0.33	5.95	86.93	5.30	1.17	0.12	0.18
BB	0.03	0.14	0.67	7.73	80.53	8.84	1.00	1.06
B	0.00	0.11	0.24	0.43	6.48	83.46	4.07	5.21
CCC	0.22	0.00	0.22	1.30	2.38	11.24	64.86	19.78
Default	0.00	0.00	0.00	0.00	0.00	0.00	0.00	100.00

Table 3: 1-year transition matrix

The sectorial default times correlations are:

	Construction	Consumer goods	Services
Construction	0.50	0.20	0.30
Consumer goods	0.20	0.60	0.34
Services	0.30	0.34	0.40

Table 4: Sector correlation matrix

4.1 Obtaining the survival functions

We use appendix A.1 to compute the survivals functions with a time resolution of 1 month. We scale the transition matrix to 1 month doing $T_{\frac{1}{12}} = T_1^{\frac{1}{12}}$. Caution, do calculations using absolute values, not percentages. 1-month transition matrix, $T_{\frac{1}{12}}$:

$$\begin{pmatrix} 99.1972 & 0.7588 & 0.0320 & 0.0020 & 0.0112 & -0.0011 & -0.0001 & 0.0000 \\ 0.0635 & 99.1745 & 0.7083 & 0.0392 & 0.0015 & 0.0120 & 0.0018 & -0.0007 \\ 0.0074 & 0.2057 & 99.1980 & 0.5102 & 0.0557 & 0.0189 & -0.0001 & 0.0042 \\ 0.0015 & 0.0239 & 0.5507 & 98.8021 & 0.5164 & 0.0871 & 0.0077 & 0.0107 \\ 0.0027 & 0.0113 & 0.0396 & 0.7581 & 98.1558 & 0.8774 & 0.0889 & 0.0664 \\ -0.0007 & 0.0098 & 0.0196 & 0.0111 & 0.6447 & 98.4443 & 0.4463 & 0.4249 \\ 0.0233 & -0.0023 & 0.0170 & 0.1287 & 0.2166 & 1.2298 & 96.4213 & 1.9657 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 100.0000 \end{pmatrix}$$

Note negative values in the computed 1-month transition matrix. The cause is that the original matrix T_1 is not a regular Markov matrix. In these cases ccruncher regularizes the matrix applying the algorithm described in appendix A.2. To simplify this example we replace negative values by 0.

We compute the survivals values at month 1 using the 1-month transition matrix (d subindex means default column):

$$S(., 1 \text{ month}) = \vec{1} - (T_{1/12})_{.d} = \begin{pmatrix} 100 \\ 100 \\ 100 \\ 100 \\ 100 \\ 100 \\ 100 \end{pmatrix} - \begin{pmatrix} 0.0000 \\ 0.0000 \\ 0.0042 \\ 0.0107 \\ 0.0664 \\ 0.4249 \\ 1.9657 \end{pmatrix} = \begin{pmatrix} 100.000 \\ 100.000 \\ 99.996 \\ 99.989 \\ 99.934 \\ 99.975 \\ 98.034 \end{pmatrix}$$

We compute the survivals values at month 2 doing:

$$S(., 2 \text{ months}) = \vec{1} - (T^2_{1/12})_{.d} = \begin{pmatrix} 100 \\ 100 \\ 100 \\ 100 \\ 100 \\ 100 \\ 100 \end{pmatrix} - \begin{pmatrix} 0.0000 \\ 0.0000 \\ 0.0084 \\ 0.0221 \\ 0.1372 \\ 0.8524 \\ 3.8664 \end{pmatrix} = \begin{pmatrix} 100.000 \\ 100.000 \\ 99.992 \\ 99.978 \\ 99.863 \\ 99.148 \\ 96.134 \end{pmatrix}$$

We compute the survivals values at month 3 doing:

$$S(., 3 \text{ months}) = \vec{1} - (T^3_{1/12})_{.d} = \begin{pmatrix} 100 \\ 100 \\ 100 \\ 100 \\ 100 \\ 100 \\ 100 \end{pmatrix} - \begin{pmatrix} 0.0000 \\ 0.0000 \\ 0.0129 \\ 0.0342 \\ 0.2122 \\ 1.2821 \\ 5.7045 \end{pmatrix} = \begin{pmatrix} 100.000 \\ 100.000 \\ 99.987 \\ 99.966 \\ 99.788 \\ 98.718 \\ 94.296 \end{pmatrix}$$

We repeat the same procedure for the 360 months to obtain the survival functions exposed at table 5.

Month	AAA	AA	A	BBB	BB	B	CCC
1	100.000	100.000	99.996	99.989	99.934	99.575	98.034
2	100.000	100.000	99.992	99.978	99.863	99.148	96.134
3	100.000	100.000	99.987	99.966	99.788	98.718	94.296
4	100.000	100.000	99.983	99.953	99.709	98.286	92.518
5	100.000	100.000	99.978	99.939	99.626	97.853	90.798
6	100.000	100.000	99.973	99.925	99.539	97.418	89.134
7	100.000	100.000	99.968	99.909	99.448	96.981	87.525
8	100.000	100.000	99.963	99.893	99.353	96.544	85.967
9	100.000	100.000	99.957	99.876	99.255	96.106	84.460
10	100.000	100.000	99.952	99.858	99.153	95.668	83.000
11	100.000	100.000	99.946	99.840	99.048	95.229	81.588
12	100.000	100.000	99.940	99.820	98.940	94.790	80.220
13	100.000	99.999	99.934	99.800	98.828	94.351	78.896
14	100.000	99.998	99.928	99.778	98.714	93.912	77.613
15	100.000	99.997	99.921	99.756	98.596	93.474	76.370
...	...	...	...	...	...	...	...
169	99.213	97.964	95.345	88.888	72.479	50.213	28.155
170	99.200	97.936	95.292	88.793	72.333	50.062	28.073
171	99.187	97.908	95.240	88.698	72.188	49.912	27.992
172	99.173	97.880	95.187	88.604	72.043	49.764	27.911
173	99.159	97.851	95.134	88.509	71.899	49.616	27.832
...	...	...	...	...	...	...	...
308	95.906	92.251	86.516	76.085	56.684	35.993	20.548
309	95.871	92.198	86.445	75.999	56.596	35.924	20.511
310	95.837	92.145	86.375	75.913	56.509	35.855	20.474
311	95.801	92.093	86.304	75.828	56.423	35.787	20.437
312	95.766	92.040	86.234	75.742	56.336	35.719	20.400
...	...	...	...	...	...	...	...
358	93.986	89.478	82.955	71.932	52.638	32.889	18.868
359	93.944	89.419	82.883	71.851	52.563	32.833	18.838
360	93.902	89.361	82.812	71.771	52.488	32.778	18.808

Table 5: Survival functions table

4.2 Creating the borrowers correlation matrix

We use section 2.1 to create the borrowers correlation matrix. In this example, the debtors are sorted by sector and therefore no reorder is required.

$$\Sigma = \left(\begin{array}{cc|cc|ccccc} 1.00 & 0.50 & 0.20 & 0.20 & 0.30 & 0.30 & 0.30 & 0.30 & 0.30 \\ 0.50 & 1.00 & 0.20 & 0.20 & 0.30 & 0.30 & 0.30 & 0.30 & 0.30 \\ \hline 0.20 & 0.20 & 1.00 & 0.60 & 0.34 & 0.34 & 0.34 & 0.34 & 0.34 \\ 0.20 & 0.20 & 0.60 & 1.00 & 0.34 & 0.34 & 0.34 & 0.34 & 0.34 \\ \hline 0.30 & 0.30 & 0.34 & 0.34 & 1.00 & 0.40 & 0.40 & 0.40 & 0.40 \\ 0.30 & 0.30 & 0.34 & 0.34 & 0.40 & 1.00 & 0.40 & 0.40 & 0.40 \\ 0.30 & 0.30 & 0.34 & 0.34 & 0.40 & 0.40 & 1.00 & 0.40 & 0.40 \\ 0.30 & 0.30 & 0.34 & 0.34 & 0.40 & 0.40 & 0.40 & 1.00 & 0.40 \\ 0.30 & 0.30 & 0.34 & 0.34 & 0.40 & 0.40 & 0.40 & 0.40 & 1.00 \end{array}\right)$$

4.3 Copula initialization

The implemented copulas, gaussian and T-Student, requires the same covariance matrix, Σ'. We compute this matrix following the steps 1 and 2 from appendices A.4 and A.5.

We create the covariance matrix Σ' mapping Σ component by component ($2sin(\rho_{ij}\frac{\pi}{6})$ transformation):

$$\Sigma' = \begin{pmatrix} 1.0000 & 0.5176 & 0.2091 & 0.2091 & 0.3129 & 0.3129 & 0.3129 & 0.3129 & 0.3129 \\ 0.5176 & 1.0000 & 0.2091 & 0.2091 & 0.3129 & 0.3129 & 0.3129 & 0.3129 & 0.3129 \\ 0.2091 & 0.2091 & 1.0000 & 0.6180 & 0.3542 & 0.3542 & 0.3542 & 0.3542 & 0.3542 \\ 0.2091 & 0.2091 & 0.6180 & 1.0000 & 0.3542 & 0.3542 & 0.3542 & 0.3542 & 0.3542 \\ 0.3129 & 0.3129 & 0.3542 & 0.3542 & 1.0000 & 0.4158 & 0.4158 & 0.4158 & 0.4158 \\ 0.3129 & 0.3129 & 0.3542 & 0.3542 & 0.4158 & 1.0000 & 0.4158 & 0.4158 & 0.4158 \\ 0.3129 & 0.3129 & 0.3542 & 0.3542 & 0.4158 & 0.4158 & 1.0000 & 0.4158 & 0.4158 \\ 0.3129 & 0.3129 & 0.3542 & 0.3542 & 0.4158 & 0.4158 & 0.4158 & 1.0000 & 0.4158 \\ 0.3129 & 0.3129 & 0.3542 & 0.3542 & 0.4158 & 0.4158 & 0.4158 & 0.4158 & 1.0000 \end{pmatrix}$$

And then we apply Cholesky to Σ', then $\Sigma' = B \cdot B^\top$, where:

$$B = \begin{pmatrix} 1.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.5176 & 0.8556 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.2091 & 0.1179 & 0.9708 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.2091 & 0.1179 & 0.5773 & 0.7805 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.3129 & 0.1764 & 0.2760 & 0.1392 & 0.8806 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.3129 & 0.1764 & 0.2760 & 0.1392 & 0.2172 & 0.8534 & 0.0000 & 0.0000 & 0.0000 \\ 0.3129 & 0.1764 & 0.2760 & 0.1392 & 0.2172 & 0.1688 & 0.8365 & 0.0000 & 0.0000 \\ 0.3129 & 0.1764 & 0.2760 & 0.1392 & 0.2172 & 0.1688 & 0.1382 & 0.8250 & 0.0000 \\ 0.3129 & 0.1764 & 0.2760 & 0.1392 & 0.2172 & 0.1688 & 0.1382 & 0.1170 & 0.8167 \end{pmatrix}$$

4.4 Monte Carlo simulation

In this example we use a gaussian copula. Each Monte Carlo trial consist of three parts, the copula simulation, the default times simulation and the portfolio loss evaluation.

Copula simulation. To simulate the gaussian copula we follows the steps 3 to 5 from appendice A.4. To simulate the multivariate $\vec{Z} \sim N(0, \Sigma')$ we simulate 9 independents $N(0,1)$ and multiplies with B.

$$\vec{Z} = B \cdot \begin{pmatrix} +0.0689 \\ -0.3096 \\ -0.2790 \\ +0.4435 \\ -1.2855 \\ -0.8553 \\ -0.6456 \\ -0.0275 \\ -0.6733 \end{pmatrix} = \begin{pmatrix} +0.0689 \\ -0.2292 \\ -0.2929 \\ +0.1630 \\ -1.1803 \\ -1.0575 \\ -1.0120 \\ -0.5839 \\ -1.1143 \end{pmatrix}$$

Finally we obtain the copula $\vec{U}$ doing:

$$\vec{U} = \begin{pmatrix} \Phi(+0.0689) \\ \Phi(-0.2292) \\ \Phi(-0.2929) \\ \Phi(+0.1630) \\ \Phi(-1.1803) \\ \Phi(-1.0575) \\ \Phi(-1.0120) \\ \Phi(-0.5839) \\ \Phi(-1.1143) \end{pmatrix} = \begin{pmatrix} 0.5275 \\ 0.4093 \\ 0.3848 \\ 0.5647 \\ 0.1189 \\ 0.1452 \\ 0.1558 \\ 0.2797 \\ 0.1326 \end{pmatrix}$$

where $\Phi(x)$ is the $N(0,1)$ cumulative distribution function.

Default times simulation. To simulate the default time we use the inverse of survivals functions as explained in section 2.2.2. The first borrower has an initial rating AA and a copula value of 52.75. In table 5 we search the month where AA is near to 52.75 and we consider this month, 961, as the month where the first borrower defaults.

Borrower	Rating	$\vec{U}_i$	Default month
B1	AA	0.5275	961
B2	BBB	0.4093	898
B3	AA	0.3848	> 1000
B4	BB	0.5647	310
B5	B	0.1189	> 1000
B6	BBB	0.1452	> 1000
B7	AAA	0.1558	> 1000
B8	CCC	0.2797	171
B9	AA	0.1326	> 1000

Table 6: Simulated default times

Portfolio loss evaluation. To evaluate the portfolio loss we use the simulated default times and the portfolio composition (tables 1 and 2) as explained in section 2.2.3.

Borrower	Default month	Asset	Exposed	Recovery	Loss
B1	961	A1		80%	
B2	898	A2		75%	
B2	898	A3		75%	
B3	> 1000	A4		60%	
B4	310	A5	70$ + 90$	90%	16$
B4	310	A6	15$	90%	1.5$
B4	310	A7		90%	
B5	> 1000	A8		30%	
B6	> 1000	A9		60%	
B7	> 1000	A10		50%	
B8	171	A11	40$ + 90$	60%	52$
B9	> 1000	A12		50%	

Table 7: Portfolio loss evaluation

The simulated portfolio loss is the sum of all asset losses, 69.5 \$ in this Monte Carlo trial.

A Appendices

A.1 From transition matrix to survival functions

The T-years transition matrix gives the probability to change from rating r_i to rating r_j in a period of T years.

$$M_T = \begin{pmatrix} m_{1,1} & \dots & m_{1,n} \\ \vdots & \ddots & \vdots \\ m_{n,1} & \dots & m_{n,n} \end{pmatrix} \qquad m_{i,j} = P(r_i \to r_j; T)$$

where n is the number of ratings and $m_{i,j}$ is the probability that a borrower with rating r_i change to rating r_j in T years. Figure 9 shows a transition matrix where the probability that a borrower with rating AA changes to rating B in 1 year is 0.14%.

	AAA	AA	A	BBB	BB	B	CCC	Default
AAA	90.81	8.33	0.68	0.06	0.12	0.00	0.00	0.00
AA	0.70	90.65	7.79	0.64	0.06	0.14	0.02	0.00
A	0.09	2.27	91.05	5.52	0.74	0.26	0.01	0.06
BBB	0.02	0.33	5.95	86.93	5.30	1.17	0.12	0.18
BB	0.03	0.14	0.67	7.73	80.53	8.84	1.00	1.06
B	0.00	0.11	0.24	0.43	6.48	83.46	4.07	5.21
CCC	0.22	0.00	0.22	1.30	2.38	11.24	64.86	19.78
Default	0.00	0.00	0.00	0.00	0.00	0.00	0.00	100.00

Figure 9: 1-year transition matrix

Transition matrix can be scaled in time using the following rules:

$$\begin{aligned} M_{T_1+T_2} &= M_{T_1} \cdot M_{T_2} \\ M_{k \cdot T} &= M_T^k \\ M_{\frac{T}{k}} &= \sqrt[k]{M_T} \end{aligned} \tag{1}$$

The root of a matrix can be computed as:

$$M = P \cdot D \cdot P^{-1} \longrightarrow M^\gamma = P \cdot D^\gamma \cdot P^{-1}$$

where P is a matrix composed by the eigenvectors of M and D is a diagonal matrix composed by the eigenvalues of M. The inverse of a matrix P can be computed using the LU decomposition of P as it is explained in *Numerical Recipes in C*[2]:

[2]http://www.nr.com

$$P \cdot P^{-1} = L \cdot U \cdot P^{-1} = Id$$

Sometimes, the scaled transition matrix don't satisfy the Markov conditions (row sum equal to 1 and all elements non-negatives). In this cases we need to transform this matrix to the close Markov matrix. This process is named regularization and is explained in appendix A.2.

So, we can compute default probability at any time and at any initial rating (in other words, the survival functions), doing:

$$Survival(r_i, t) = 1 - (M_t)_{i,n}$$

where r_i is the initial rating, t is the time, M_t is the transition matrix for time t (scaled from M_T using (1)) and n is the index of default rating, r_n.

A.2 Transition matrix regularization

The algorithm described here is extracted from [1]. The QOM (Quasi-Optimization of the root Matrix) algorithm regularize a transition matrix, m_{ij}, of dimension n row-by-row. The steps to regularize the i-th row are:

Step 1. Compute the difference betwen 1 and the row sum. Divide by n and substract this value from all non-zero components:

$$m_{ij} \neq 0 \Longrightarrow m_{ij} = m_{ij} - \frac{1}{n}\left(\sum_{j=1}^{n} m_{ij} - 1\right)$$

Step 2. If all the row elements are non-negative and sums 1 then stop, the row is regularized.

Step 3'. Fix any negative row element to zero and go to step 1.

The iterative algorithm stops after m steps where $m \leq n$ (Merkoulovitch 2000).

Apply the previous algorithm for every row. The final matrix is a regularized matrix.

A.3 Correlation matrix estimation

In this section we expose a simple method to estimate the default times correlation matrix. Supose that we dispose the number of components and the number of defaulted components for each sector for the last years (eg. 1980-2008).

Year	$Sector_1$	...	$Sector_m$
1	$a_{1,1}$	...	$a_{1,m}$
2	$a_{2,1}$	...	$a_{2,m}$
3	$a_{3,1}$	...	$a_{3,m}$
⋮	⋮	⋮	⋮
T-1	$a_{T-1,1}$	...	$a_{T-1,m}$
T	$a_{T,1}$	...	$a_{T,m}$

Year	$Sector_1$	...	$Sector_m$
1	$d_{1,1}$	...	$d_{1,m}$
2	$d_{2,1}$	...	$d_{2,m}$
3	$d_{3,1}$	...	$d_{3,m}$
⋮	⋮	⋮	⋮
T-1	$d_{T-1,1}$	...	$d_{T-1,m}$
T	$d_{T,1}$	...	$d_{T,m}$

where $a_{i,j}$ are the number of borrowers in sector j at year i and $d_{i,j}$ is the number of defaulted borrowers of sector j at year i.

We use the following definition of correlation:

$$Correl(S_i, S_j) = \frac{P(S_i \cap S_j) - P(S_i) \cdot P(S_j)}{\sqrt{P(S_i) \cdot (1 - P(S_i))} \cdot \sqrt{P(S_j) \cdot (1 - P(S_j))}}$$

Where,

$$P(S_i) = \frac{1}{T} \sum_{t=1}^{T} \frac{d_{t,i}}{a_{t,i}}$$

$$P(S_i \cap S_j) = \begin{cases} \frac{1}{T} \sum_{t=1}^{T} \frac{d_{t,i} \cdot d_{t,j}}{a_{t,i} \cdot a_{t,j}} & \text{if } i \neq j \\ \frac{1}{T} \sum_{t=1}^{T} \frac{d_{t,i} \cdot (d_{t,i} - 1)}{a_{t,i} \cdot (a_{t,i} - 1)} & \text{if } i = j \end{cases}$$

You can find more information about correlation matrix estimation on [11] and [10].

A.4 Gaussian copula simulation

Algorithm to simulate a gaussian copula satisfying correlation matrix Σ [14] [5]:

Step 1. We create the covariance[3] matrix Σ' mapping Σ component by component:

$$\Sigma' = \begin{pmatrix} 2sin(\frac{\pi}{6}) & 2sin(\rho_{12}\frac{\pi}{6}) & \dots & 2sin(\rho_{1n}\frac{\pi}{6}) \\ 2sin(\rho_{12}\frac{\pi}{6}) & 2sin(\frac{\pi}{6}) & \dots & 2sin(\rho_{2n}\frac{\pi}{6}) \\ \vdots & \vdots & \ddots & \vdots \\ 2sin(\rho_{1n}\frac{\pi}{6}) & 2sin(\rho_{2n}\frac{\pi}{6}) & \dots & 2sin(\frac{\pi}{6}) \end{pmatrix}$$

Observe that Σ' have diagonal elements with 1 because $2sin(\frac{\pi}{6}) = 1$.

Step 2. We apply Cholesky[4] to Σ', then $\Sigma' = B \cdot B^\top$, where:

$$B = \begin{pmatrix} b_{11} & 0 & \dots & 0 \\ b_{21} & b_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \dots & b_{nn} \end{pmatrix}$$

Step 3. We simulate a $N(0,1)$ n times:

$$\vec{Y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \qquad y_k \sim N(0,1) \text{ independents}$$

Step 4. We simulate a multivariate normal, $\vec{Z} \sim N(\vec{0}, \Sigma')$, doing:

$$B \cdot \vec{Y} = \begin{pmatrix} b_{11} & 0 & \dots & 0 \\ b_{21} & b_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \dots & b_{nn} \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} z_1 \\ \vdots \\ z_n \end{pmatrix} = \vec{Z}$$

Step 5. Finally we obtain the copula $\vec{U}$ doing:

$$\begin{pmatrix} \Phi(z_1) \\ \vdots \\ \Phi(z_n) \end{pmatrix} = \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} = \vec{U}$$

where $\Phi(x)$ is the $N(0,1)$ cumulative distribution function.

[3] Correlation and covariance matrix are the same because diagonal elements are 1.

[4] Appendix A.6 adapts Cholesky decomposition to deal with block matrix.

A.5 T-Student copula simulation

Algorithm to simulate a t-Sudent copula with ν degrees of freedom ($\nu > 2$) satisfying correlation matrix Σ [5]:

Step 1. We create the covariance[5] matrix Σ' mapping Σ component by component:

$$\Sigma' = \begin{pmatrix} 2sin(\frac{\pi}{6}) & 2sin(\rho_{12}\frac{\pi}{6}) & \dots & 2sin(\rho_{1n}\frac{\pi}{6}) \\ 2sin(\rho_{12}\frac{\pi}{6}) & 2sin(\frac{\pi}{6}) & \dots & 2sin(\rho_{2n}\frac{\pi}{6}) \\ \vdots & \vdots & \ddots & \vdots \\ 2sin(\rho_{1n}\frac{\pi}{6}) & 2sin(\rho_{2n}\frac{\pi}{6}) & \dots & 2sin(\frac{\pi}{6}) \end{pmatrix}$$

Observe that Σ' have diagonal elements with 1 because $2sin(\frac{\pi}{6}) = 1$.

Step 2. We apply Cholesky[6] to Σ', then $\Sigma' = B \cdot B^{\top}$, where:

$$B = \begin{pmatrix} b_{11} & 0 & \dots & 0 \\ b_{21} & b_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \dots & b_{nn} \end{pmatrix}$$

Step 3. We simulate a $N(0,1)$ n times:

$$\vec{Y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \qquad y_k \sim N(0,1) \text{ independents}$$

Step 4. We simulate a chi-square with ν degrees of freedom independent of $\vec{Y}$:

$$s \sim \chi^2(\nu)$$

Step 5. We simulate a multivariate t-Student with ν degrees of freedom, $\vec{Z} \sim \frac{\sqrt{\nu}}{\sqrt{\chi^2(\nu)}} \cdot N(\vec{0}, \Sigma')$, doing:

$$\frac{\sqrt{\nu}}{\sqrt{s}} \cdot B \cdot \vec{Y} = \frac{\sqrt{\nu}}{\sqrt{s}} \cdot \begin{pmatrix} b_{11} & 0 & \dots & 0 \\ b_{21} & b_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \dots & b_{nn} \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} z_1 \\ \vdots \\ z_n \end{pmatrix} = \vec{Z}$$

[5] Correlation and covariance matrix are the same because diagonal elements are 1.
[6] Appendix A.6 adapts Cholesky decomposition to deal with block matrix.

Step 6. Finally we obtain the copula $\vec{U}$ doing:

$$\begin{pmatrix} t_\nu(z_1) \\ \vdots \\ t_\nu(z_n) \end{pmatrix} = \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} = \vec{U}$$

where $t_\nu(x)$ is the t-Student with ν degrees of freedom cumulative distribution function.

A.6 Cholesky decomposition for symmetric block matrix

Cholesky algorithm decomposes a symmetric positive definite matrix into a lower triangular matrix and the transpose of the lower triangular matrix. Algorithm description can be found in *Numerical Recipes in C*[7].

$$A = U^\top \cdot U$$

If we have a portfolio of 50000 borrowers, the correlation matrix size will be a 50000×50000. This requires up to 19 GB of RAM memory. The multiplication of this matrix by a vector implies 2500000000 multiplications. This is far too much. So we adapt the Cholesky algorithm in order to consider that the borrowers' correlation matrix is a block matrix with 1's in diagonal. For instance:

$$A = \left(\begin{array}{cccc|ccc} 1 & 0.5 & 0.5 & 0.5 & 0.1 & 0.1 & 0.1 \\ 0.5 & 1 & 0.5 & 0.5 & 0.1 & 0.1 & 0.1 \\ 0.5 & 0.5 & 1 & 0.5 & 0.1 & 0.1 & 0.1 \\ 0.5 & 0.5 & 0.5 & 1 & 0.1 & 0.1 & 0.1 \\ \hline 0.1 & 0.1 & 0.1 & 0.1 & 1 & 0.3 & 0.3 \\ 0.1 & 0.1 & 0.1 & 0.1 & 0.3 & 1 & 0.3 \\ 0.1 & 0.1 & 0.1 & 0.1 & 0.3 & 0.3 & 1 \end{array}\right)$$

We decompose the previous matrix using the standard Cholesky decomposition:

$$U = \left(\begin{array}{cccc|ccc} 1.00000 & 0.50000 & 0.50000 & 0.50000 & 0.10000 & 0.10000 & 0.10000 \\ 0 & 0.86603 & 0.28868 & 0.28868 & 0.05774 & 0.05774 & 0.05774 \\ 0 & 0 & 0.81650 & 0.20412 & 0.04082 & 0.04082 & 0.04082 \\ 0 & 0 & 0 & 0.79057 & 0.03162 & 0.03162 & 0.03162 \\ \hline 0 & 0 & 0 & 0 & 0.99197 & 0.28630 & 0.28630 \\ 0 & 0 & 0 & 0 & 0 & 0.94975 & 0.21272 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.92563 \end{array}\right)$$

[7] http://www.nr.com

We can see that U have repeated elements that can be kept in RAM memory in this way:

$$U = \left|\begin{array}{c|cc} 1.00000 & 0.50000 & 0.10000 \\ 0.86603 & 0.28868 & 0.05774 \\ 0.81650 & 0.20412 & 0.04082 \\ 0.79057 & 0 & 0.03162 \\ 0.99197 & 0 & 0.28630 \\ 0.94975 & 0 & 0.21272 \\ 0.92563 & 0 & 0 \end{array}\right|$$

that is, for each row we keep the diagonal value and the value of each sector. With this strategy the required memory size is $N \times (M + 1)$ where N is the number of borrowers and M the number of sectors. With this consideration the memory required to store a 50000×50000 borrowers correlation matrix is only 4.2 Mb.

We use the fact that matrix U have repeated elements to reduce the number of operations required to multiply U by a vector. Let see an example:

$$\begin{aligned} (U \cdot x)_2 &= 0.0 \cdot x_1 + 0.86603 \cdot x_2 + 0.28868 \cdot x_3 + 0.28868 \cdot x_4 + \\ &\quad 0.05774 \cdot x_5 + 0.05774 \cdot x_6 + 0.05774 \cdot x_7 \\ &= 0.86603 \cdot x_2 + 0.28868 \cdot (x_3 + x_4) + 0.05774 \cdot (x_5 + x_6 + x_7) \end{aligned}$$

Considering this, we can reduce the number of operations from N^2 to $N \times (M + 1)$ where N is the number of borrowers and M the number of sectors. In the 50000 borrowers example, the operations number reduces from 2500000000 to only 500000.

A.7 Condition number for symmetric block matrix

The condition number of a matrix indicates how far or close is to be a singular matrix. If the condition number number is close to 1 is said that it is well-conditioned while a matrix with a high condition number is said to be ill-conditioned.

Given a symmetric block matrix withs 1's in the diagonal, Σ, we desire to obtain the condition number of the cholesky decomposition matrix, B. Using condition number definition and its properties we have that:

$$\kappa(B) = \|B\|_2 \cdot \|B^{-1}\|_2 = \frac{\sigma_{max}(B)}{\sigma_{min}(B)} = \sqrt{\frac{\lambda_{max}(\Sigma)}{\lambda_{min}(\Sigma)}}$$

Where $\sigma_{max}(B)$ and $\sigma_{min}(B)$ are the maximal and minimal singular values of B and $\lambda_{max}(\Sigma)$ and $\lambda_{min}(\Sigma)$ are the maximal and minimal absolute values of eigenvalues of Σ. The eigenvalues of Σ can be computed using the following proposition.

Proposition (unproved). Let Σ a symmetric block matrix with 1's in the diagonal where k is the number of blocks, a_{ij} are the block values and n_i is the dimension of the i-th block. Then, the eigenvalues of Σ are:

$$\underbrace{1-a_{11},\cdots,1-a_{11}}_{n_1-1},\underbrace{1-a_{22},\cdots,1-a_{22}}_{n_2-1},\cdots\underbrace{1-a_{kk},\cdots,1-a_{kk}}_{n_k-1},\lambda_1,\cdots,\lambda_k$$

Where λ_i are the eigenvalues of the following deflated matrix:

$$K=\begin{pmatrix} 1+(n_1-1)\cdot a_{11} & n_2\cdot a_{12} & \cdots & n_k\cdot a_{1k} \\ n_1\cdot a_{12} & 1+(n_2-1)\cdot a_{22} & \cdots & n_k\cdot a_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ n_1\cdot a_{1k} & n_2\cdot a_{2k} & \cdots & 1+(n_k-1)\cdot a_{kk} \end{pmatrix}$$

References

[1] Marina Sidelnikova Alexander Kreinin. Regularization algorithms for transition matrices. *Algo Research Quarterly, Vol. 4, Nos. 1/2*, 2001.

[2] Dirk Tasche Carlo Acerbi. Expected shortfall: a natural coherent alternative to value at risk. *BIS*, 2001.

[3] Darrel Duffie and Kenneth J. Singleton. *Credit Risk. Pricing, Measurement, and Management*. Princeton University Press, 2003.

[4] Marno Verbeek Erik Kole, Kees Koedijk. Selecting copulas for risk management. 2006.

[5] Elisabeth Joossens Francesco Saita, Francesca Campolongo and Maria Egle Romano. Pricing multiasset equity options with copulas: an empirical test.

[6] Greg M. Gupton, Christopher C. Finger, and Mickey Bhatia. *CreditMetrics - Technical Document*. J.P. Morgan & Co. Incorporated, 1997.

[7] Jonathan E. Grindlay Jaesub Hong, Eric M. Schlegel. New spectral classification technique for x-ray sources: quantile analysis. 2004.

[8] Stefan R. Jaschke. Quantile-var is the wrong measure to quantify market risk for regulatory purposes. *BIS*, 2001.

[9] Philippe Jorion. *Value at Risk*. McGraw-Hill, 1997.

[10] David X. Li. *On Default Correlation: A Copula Function Approach*. The RiskMetrics Group, 2000.

[11] Douglas Lucas. *Default Correlation: From Definition to Proposed Solutions*. CDO Research, 2004.

[12] Alexander McNeil Paul Embrechts and Daniel Straumann. Correlation and dependence in risk management: properties and pitfalls. *RiskLab*, 1999.

[13] Murray R. Spiegel. *Estadística*. Schaum, 1997.

[14] Shaun S. Wang. Aggregation of correlated risk portfolios: Models & algorithms. *CAS Committee on Theory of Risk*.