Quantitative Risk Management
Final Assignment

John Dodson

December 12, 2020

This assignment is not a regular homework. It is worth half of the module grade for my module. It is due in Canvas at 9:00 PM Central Time on December 23, 2020. You are welcome to discuss the project with our teaching assistant, but she will not be grading it. I will hold regular office hours starting at 7:00 PM on Sunday, December 13 and 20 and I will try to be available if you want to discuss outside of office hours.

If you discuss this assignment with anyone other than the instructor, please summarize those collaborations in an acknowledgement statement in your introduction.

Normal Mixture Copulas

Normal mixture copulas are characterized by a generator, with possibly several parameters, and a collation of pairwise "concordances" in the form of a symmetric positive semi-definite matrix with unit diagonal entries.

One way to simulate uniform draws from a normal mixture copula is to convert draws from the corresponding mixture. To simulate draws from the normal variance mixture, draw correlated normals and scale them by the square root of draws from the variance mixing distribution. N.B.: Apply the same variance draw for each of the components.

In the case of a t_ν copula, the variances are mixed by an inverse Gamma random variable. If ν is a positive integer then

$$\frac{\nu}{Z_1^2 + \cdots + Z_\nu^2} \sim \text{Gamma}^{-1}\left(\frac{\nu}{2}, \frac{\nu}{2}\right)$$

for i.i.d. $Z_i \sim \mathcal{N}(0,1)$,

The marginals are Student's-t_ν, so to get the equivalent copula draws apply the marginal distribution functions. For small integer ν, there are closed-form expressions for these, e.g. for $\nu = 4$,

$$F(x; \nu = 4) = \frac{1}{2}\left(1 + \frac{x\left(6 + x^2\right)}{(4 + x^2)^{3/2}}\right)$$

Normal Reciprocal Inverse Gaussian Residuals

An example of a leptokurtotic random variable from the Gumbel extreme value class is the symmetric normal reciprocal inverse Gaussian ("NRIG"), which is a special case of the generalized hyperbolic normal variance mixture. The density of the standard version can be written as

$$f(x; g) = \frac{1}{\pi} e^g K_0\left(\sqrt{g^2 + (1+g)x^2}\right)\sqrt{1+g}$$

where K is Bessel's function of the third kind, and the parameter g is positive.

Exploratory analysis of the standardized log return residuals for Treasury bonds from the last homework suggests that they are approximately symmetric but clearly leptokurtotic. These features can be modeled by NRIG random variables. The maximum likelihood fits for the shape parameters of the residuals are tabulated in 1.

tenor	$\hat{g}$
1Y	1.03
2Y	2.43
3Y	2.70
5Y	3.04
7Y	3.79
10Y	4.68
20Y	3.66
30Y	3.63

Table 1: Maximum likelihood NRIG shape parameters for CMT residuals

As an aside, recall that large g in this setting is consistent with diffusion and small g is suggestive of jumps in the stochastic process. So it is satisfying to note that the largest $\hat{g}$ we observe is in the ten-year, which is also the most actively traded in part because of its importance to hedging the interest rate exposure of mortgages; while the smallest $\hat{g}$ we observe is in the one-year, which is also the most sensitive to the discreteness of Federal Reserve rate setting policy.

Simulation

In order to convert copula draws into standard NRIG draws we must apply the marginal quantile function, which is also called the probability integral transform. There is no closed form expression for this, but Newton-Raphason is efficient enough[1].

In order to evaluate the quantile at p, look for the root of the equation

$$h(x) = \tfrac{1}{2} - p + \int_0^x f(x'; g)\, dx'$$

Note that this version takes advantage of the symmetry of the model to make the integral definite.

Since $h'(x) = f(x; g)$, iterate on

$$x_{i+1} \leftarrow x_i - \left(\tfrac{1}{2} - p + \int_0^{x_i} f(x'; g)\, dx' \right) / f(x_i; g)$$

This seems to converge to within machine precision within four or five iterations.

Problems

In the last homework, we identified three orthogonal portfolios of Treasury securities, along with forecasts for the one-day standard deviations of each Treasury log-return and corresponding pairwise concordances.

With the NRIG parameterizations above, we have all we need to assemble the joint distribution for the log-returns of financed Treasury positions for November 30, 2020, under any normal mixture copula.

1. Using simulation, create 10,000 draws for the profit/loss per dollar notional in each of the CMT tenors under two copulas: the Gaussian copula and a t_4 copula.

   Start with copula draws, then convert to marginal residuals, then scale to get log returns and exponentiate to get prices, then subtract to get profit/loss.

2. Calculate the profit/loss of each of the the three portfolios for each scenario.

3. Interpreting these six arrays as empirical loss distributions, calculate the 99% one-day value-at-risk and expected shortfall for each portfolio for each copula and interpret the results.

[1] This can be very computationally expensive and is typically the rate limiting step in simulation-based loss modeling.

Grading Rubric

Twenty out of fifty points will be based on the follow criteria:

- Your report is clear and professional, and includes an introduction and conclusion. **(5 points)**
- Your analyses are well-reasoned and convincing. **(5 points)**
- You include or attach your scripts and they are documented. **(5 points)**
- You include appropriate citations and acknowledgements. **(5 points)**

Solution

Please refer to the December 2 and December 9 exercises for background on the problem set-up, data acquisition, model calibration, and portfolio selection. In this capstone project, we are evaluating some loss distribution risk measures and exploring in particular the infleunce of the tail dependence assumption.

Copula draws

Since simulation may involve a large number of pseudo-random variates, and it is useful for applications to be repeatable, it is important to spend a moment thinking about your random number generator. In a single-threaded environment, you can depend on the generator to maintain its own state between calls, which I do below. With distributed processing you may need to draw a sequence ahead of time and distribute ranges of it to the different threads.

The main algorithm today for long-period pseudo-random numbers for simulation modeling is the "Mersenne Twister" developed by Matsumoto and Nishimura. The seed is typically a `UInt32` or an array of `UInt32`. By fixing the seed, the sequence returned will be fixed. Obviously the value of the seed should not affect your application of the results—if it does then your simulation size is probably not large enough.

```
"pseudo-random number generator to use for simulation"
rng = MersenneTwister(0x00005031);
```

Since we are working with elliptical copulas, we start the simulation with a sample of correlated standard normals. Here N is the number of scenarios and `P` (capital rho) is the (symmetric, positive-definite, unit-diagonal) $d \times d$ concordance matrix for the $d = 8$ CMT tenors we are working with.

```
"simulation size"
N = 10_000

"correlated normal variates"
Z_variates = randn(rng,N,size(P,1))*cholesky(P).U
```

I am multiplying on the right by the upper-diagonal version of the Cholesky decomposition so that my result has shape $N \times d$ and functions such as `cov` work naturally.

Since the margins are standard normal, we can apply the standard normal distribution function to convert to uniform grades.

```
"standard normal distribution function"
Gauss_dist = x -> (1+erf(x/sqrt(2)))/2

"Gaussian copula draws"
Gauss_draws = Gauss_dist.(Z_variates)
```

I checked the summary statistics of the results just to make sure the draws were reasonable.

```
minimum(Gauss_draws,dims=1)     # should each be slightly above zero
maximum(Gauss_draws,dims=1)     # should each be slightly below one
mean(Gauss_draws,dims=1)        # should each be close to one-half
var(Gauss_draws,dims=1)         # should each be close to one-twelfth
```

For the t_4 copula, we have to scale the correlated normals in each scenario by the square root of a draw from the variance mixing distribution.

```
"correlated Student's-t variates"
t_variates = Z_variates.*sqrt.(4 ./sum(randn(rng,N,4).^2,dims=2))
```

Apply the Student's-t_4 distribution function to get copula draws.

```
"Student's-t_4 distribution function"
t4_dist = x -> (1+x*(6+x^2)/(4+x^2)^(3/2))/2

"t_4 copula draws"
t_draws = t4_dist.(t_variates)
```

The summary statistics on these looked reasonable, too.

A sample of the copula draws is plotted in Figure 1. Notice that the t_4 copula, which has non-zero tail dependence, shows more clustering along the diagonal for extreme draws.

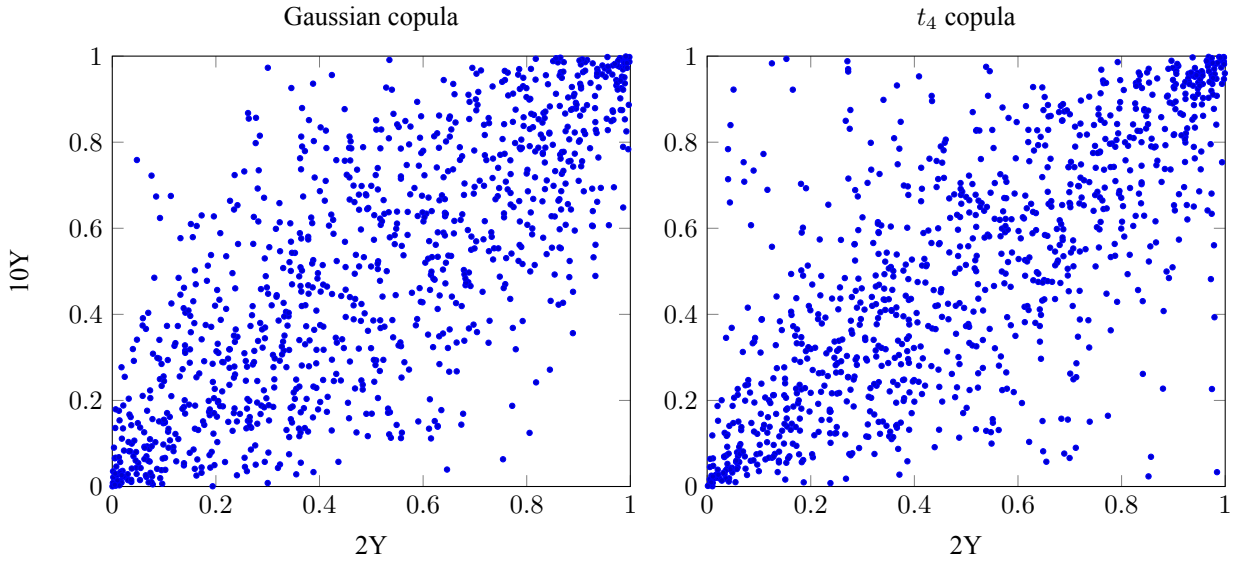

Figure 1: Examples of copula draws

Log-return scenarios

To convert the copula draws to standardized log-returns, apply the marginal quantile functions. In the introduction, I described an algorithm for calculated the quantile at a point, p, for the density function. For the NRIG, the density is implemented as

```
"standard NRIG density"
function NRIG_dens(x,g)
    return exp(g)sqrt(1+g)besselk(0,sqrt(g^2+(1+g)*x^2))/π
end
```

Note that most special functions libraries use some version of Donald Amos' "algorithm 644", originally written in FORTAN77, for evaluating Bessel functions. In Julia, the function $(\nu, x) \mapsto K_\nu(x)$ is implemented in `besselk` used above.

The quantile function is implemented as

```
function NRIG_quan(p,g)
# Newton-Raphson method for standard symmetric NRIG probability integral transform
    x0 = 0. # initial guess
    x1 = NaN
    for n = 1:20
        f = NRIG_dens(x0,g)
        x1 = x0-(1/2-p+quadgk(x -> NRIG_dens(x,g),0,x0)[1])/f
        if abs(x1-x0) < 1e-12
            break
        end
        x0 = x1
    end
    return x1
end
```

Note that `quadgk` uses an adaptive fifteen-point Gauss-Kronod algorithm for numerical integration (quadrature). This is effective when the integrand is smooth, which is the case here. I am using a maximum of twenty iterations, which seems to work well for `p` between about `1E-7` and `1-1E-7` for `g = 1.` or greater. This is sufficient for our purposes, but it could probably be improved by bringing the $1/f(x_i; g)$ into the integrand and using the alternate `besselkx` in the numerator and denominator instead.

The marginal log-returns each have different distributions, at least different parameterizations. In my implementation, those parameterizations are in the arrays `g` and `σ`.

```
"log-return scenarios for the Gaussian copula"
Gauss_logret = NRIG_quan.(Gauss_draws,g').*σ'

"log-return scenarios for the t_4 copula"
t_logret = NRIG_quan.(t_draws,g').*σ'
```

The quantile transformation can be time-consuming. In this case, it entails about a million numerical integrations, each of while entails at least fifteen `besselk` evaluations. Thankfully, since Julia compiles to LLVM intermediate representation and we only have eight components, this took only about eight seconds to run on my third generation MacBook Pro. I have heard reports of similar timings for python3 (which is interpreted at runtime) on newer hardware.

Profit/loss scenarios

Converting log-return scenarios to profit/loss per unit of notional is straight-forward.

```
"simple return scenarios for the Gaussian copula"
Gauss_scen = exp.(Gauss_logret).-1

"simple return scenarios for the t_4 copula"
t_scen = exp.(t_logret).-1
```

This is the step that relies crucially on the Gumbel extreme value class right tail. If we were using a Fréchet class right tail for the log-return, such as a symmetric Student's-t, simple returns would not be usable for calculating the expected shortfall on a short position because the result would diverge in the simulation size.

I have assembled the arrays of notional quantities for the three portfolios we are analyzing in arrays `w₁`, `w₂`, and `w₃`.

The empirical distribution of the loss exceeding 99%,

```
"loss r.v. confidence level"
α = 0.99

"points in the loss tail"
N₀ = floor(Int,N*(1-α)+1/2)
```

is:

```
t_p₁ = partialsort!(t_scen*w₁,1:N₀)
t_p₂ = partialsort!(t_scen*w₂,1:N₀)
t_p₃ = partialsort!(t_scen*w₃,1:N₀)

Gauss_p₁ = partialsort!(Gauss_scen*w₁,1:N₀)
Gauss_p₂ = partialsort!(Gauss_scen*w₂,1:N₀)
Gauss_p₃ = partialsort!(Gauss_scen*w₃,1:N₀)
```

Notice that I am using partial sort rather then full sort, because we only care about the extremes. This is about five times faster for identifying 100 out of 10,000 draws. Efficiencies like this are worth taking when you are developing a realtime risk monitoring system, for example.

Risk measures

In the last step, we evaluate the risk measures. The value-at-risk estimate is the least extreme of the extreme losses. The expected shortfall is the average of the extreme losses.

```
t_ES₁ = mean(t_p₁); t_ES₂ = mean(t_p₂); t_ES₃ = mean(t_p₃)
t_VaR₁ = t_p₁[end]; t_VaR₂ = t_p₂[end]; t_VaR₃ = t_p₃[end]

Gauss_ES₁ = mean(Gauss_p₁); Gauss_ES₂ = mean(Gauss_p₂); Gauss_ES₃ = mean(Gauss_p₃)
Gauss_VaR₁ = Gauss_p₁[end]; Gauss_VaR₂ = Gauss_p₂[end]; Gauss_VaR₃ = Gauss_p₃[end]
```

Results are presented in Table 2.

ES	t_4	Gauss	diff.
1st	-1516.	-1380.	10%
2nd	-314.	-278.	13%
3rd	-240.	-192.	25%

VaR	t_4	Gauss	diff.
1st	-1242.	-1169.	6%
2nd	-242.	-227.	7%
3rd	-181.	-159.	14%

Table 2: One-day 99% value-at-risk and expected shortfall per million gross notional in Treasuries as of November 27, 2020

Discussion

It is not surprising that the first portfolio, which was long every CMT tenor, has the largest risk per unit of gross notional. Clearly all of the tenors are highly concordant. Many buy-side bond managers feel little need to measure risk beyond duration, which—like CAPM beta—is a heuristic measure of sensitivity to systematic risk.

But clearly there are other material risks once you consider portfolios with spreads such as the second and the third. And to begin to assess these other risks we have to make some assumptions, as exhibited by this project.

Amongst these assumptions, tail dependence is one of the more subtle and difficult to explain and justify. In some settings, the choice of copula may not be important. But I find its influence well-demonstrated in this project. Surveying again Table 2, it seems that the copula assumption is not particularly influential for pure duration exposures or even curve steepen/flatten exposures as in the second portfolio—particularly for value-at-risk. But when the exposures start

to become more complicated, as in the third portfolio, tail dependence becomes much more influential—particularly for expected shortfall.

A firm with a strong control culture may be inclined to enforce hard risk limits. There may be an element of hubris to this if the metrics behind those limits are not sufficiently realistic. Consider, for example, Treasury desk trading limits based on duration or value-at-risk. We have already discussed Claudio Albanese's criticism of value-at-risk for defaultable bonds. Here is an example of where value-at-risk fails for non-defaultable bonds if, in fact, there is material tail dependence. Traders under duration or value-at-risk limits are effectively encouraged to "hedge" their systematic risks and lever up on more subtle "basis" exposures. There would be less incentive for this under an expected shortfall limit, under a risk model with a conservative assumption about tail dependence.