Quantitative Risk Management
Final Project

John Dodson

December 18, 2019

This assignment is not a regular homework. It is a group project worth half of the module grade for the fall term. If you discuss this assignment with anyone other than the instructor, please summarize those discussions in a statement acknowledging your collaborations.

Please e-mail or share your solution with me before 5:30 PM on Wednesday, December 18. Please turn in your report directly to me. You are welcome to discuss the project with our TA, but she will not be grading it.

Introduction

For this project, I would like you to return to the analysis of the NASDAQ-100 index options we worked with previously. For the October 30 assignment, you fit—and rejected—a normal to the implied distribution of the logarithm of the terminal value of the index. For the November 13 assignment, you fit generalized Pareto distributions to the left and right tails of the logarithm, and we discussed the consequence of the apparently Fréchet right tail.

We have since learned about variance mixtures of normals, including the Student's-t, which is the basis for many empirical examples in the text. The symmetric version is probably not useful in the risk-neutral setting, but the asymmetric version might be. In this project, I would like you to produce a report discussing the applicability of this distribution to approximating index option prices.

Skewed Student's-t

Let's recall the definition of the Generalized Hyperbolic version of the skewed Student's-t random variable.

Let Q be a random variable with positive support. A Normal mixture model can be specified as

$$X|Q \sim \mathcal{N}(\mu + \beta Q, Q)$$

in terms of which

$$f_X(x) = \int_0^\infty \frac{1}{\sqrt{2\pi q}} e^{\frac{-1}{2q}(x-\mu-\beta q)^2} f_Q(q)\, dq$$

where the mixing variable is a Reciprocal Gamma,

$$f_Q(q) = \frac{\left(\frac{\chi}{2}\right)^{\frac{\nu}{2}}}{\Gamma\left(\frac{\nu}{2}\right)} q^{-\frac{\nu}{2}-1} e^{-\frac{\chi}{2q}} \quad \text{for } \chi > 0,\ \nu > 2$$

Problem Statement

Please produce a report or presentation explaining the problems below, your solution approach, intermediate and final results, and potential interpretations. Please include your complete code either inline or in an appendix.

1. Evaluate the integral $\mathrm{E}\left[e^X\right]$ in terms of the (real) parameters χ, ν, β and μ and note any conditions required for the result to be finite. In the subsequent statistical exercise, we can use this result to eliminate μ from the fit by fixing this value to the implied forward of the index F_t^T. **(10 points)**

2. Using the technique we developed for the November 13 assignment—which you should describe in your report—fit the remaining parameters, χ, ν, and β, within the constraints defined above, to the index options prices from October 28, 2019. **(10 points)**

3. Evaluate the Black-Scholes implied volatilities for both the original data and the model you have fit and plot them versus log-moneyness[1]. **(10 points)**

Hints: For (1.) write down the expectation in terms of the more general inverse gaussian mixing variable and complete the square in the exponent. For (2.) you already have the distribution function for the standardized skewed Student's-t. You also have formulas for the mean and variance of the generalized hyperbolic in the same location-scale family. So you can construct the distribution function for X with an affine transformation, and you can numerically integrate it to get the distribution function. For (3.) use the model to evaluate theoretical put prices based on $\mathrm{E}\left[\max\left(0, K - e^X\right)\right]$ from which you can get implied volatilities by inverting Black-Scholes in the usual manner.

Grading Rubric

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

- Your report or presentation is clear and professional. **(5 points)**
- Your derivations are clear and complete. **(5 points)**
- Your code is clear and documented. **(5 points)**
- You include appropriate citations and collaboration ackowledgements. **(5 points)**

Solution

Fixing the expectation

We will be using this model to price European-style index options. One of key tenants of arbitrage-free valuation of financial derivatives is that the forward price F_t^T is identified with the (risk-neutral) expected terminal value $S_T(\omega)$ for $\omega \in \mathcal{F}_t$. Since we are identifying the skewed Student's-t r.v. X with the logarithm of this value, the expectation, conditioned on the present sigma algebra, is

$$F_t^T = \mathrm{E}\left[e^X\right] = \int_{-\infty}^{\infty}\int_0^{\infty} e^x \cdot \frac{1}{\sqrt{2\pi q}} e^{\frac{-1}{2q}(x-\mu-\beta q)^2} \frac{\left(\frac{\chi}{2}\right)^{\frac{\nu}{2}}}{\Gamma\left(\frac{\nu}{2}\right)} q^{-\frac{\nu}{2}-1} e^{-\frac{\chi}{2q}}\, dq\, dx$$

[1] log-moneyness is $\log\left(K/F_t^T\right)$ in terms of the strike price K

Completing the square in the exponent, we see that

$$e^x \cdot e^{\frac{-1}{2q}(x-\mu-\beta q)^2} = e^{\mu} \cdot e^{\frac{-1}{2q}(x-\mu-(\beta+1)q)^2} \cdot e^{q\left(\beta+\frac{1}{2}\right)}$$

but from the density of the generalized hyperbolic r.v. with $\psi' = -2\beta - 1$ and $\beta' = \beta + 1$ we know that

$$1 = \int_{-\infty}^{\infty} \int_0^{\infty} \frac{1}{\sqrt{2\pi q}} e^{\frac{-1}{2q}(x-\mu-\beta' q)^2} \frac{\chi^{\frac{\nu}{2}} \left(\sqrt{\chi\psi'}\right)^{-\frac{\nu}{2}}}{2\, K_{\frac{\nu}{2}}\left(\sqrt{\chi\psi'}\right)} q^{-\frac{\nu}{2}-1} e^{-\frac{\chi}{2q}-\frac{q\psi'}{2}}\, dq\, dx$$

where $\psi' > 0$ requires $\beta < -\frac{1}{2}$. So

$$F_t^T = e^{\mu} \frac{2\left(\frac{\chi}{2}\right)^{\frac{\nu}{2}}}{\Gamma\left(\frac{\nu}{2}\right)\chi^{\frac{\nu}{2}}} K_{\frac{\nu}{2}}\left(\sqrt{\chi\left(-2\beta-1\right)}\right)\left(\sqrt{\chi\left(-2\beta-1\right)}\right)^{\frac{\nu}{2}}$$

or

$$\mu = \log\left(F_t^T\right) - \log\left(\frac{2}{\Gamma\left(\frac{\nu}{2}\right)} K_{\frac{\nu}{2}}\left(\sqrt{\chi\left(-2\beta-1\right)}\right)\left(\tfrac{1}{2}\sqrt{\chi\left(-2\beta-1\right)}\right)^{\frac{\nu}{2}}\right)$$

Since we can reset the numéraire to be the forward price, and the logarithm of one is of course zero, I use this definition henceforth:

$$\mu(\chi,\beta,\nu) = \mu_0 = -\log\left(\frac{2}{\Gamma\left(\frac{\nu}{2}\right)} K_{\frac{\nu}{2}}\left(\sqrt{\chi\left(-2\beta-1\right)}\right)\left(\tfrac{1}{2}\sqrt{\chi\left(-2\beta-1\right)}\right)^{\frac{\nu}{2}}\right)$$

The full density function of the skewed-Student's-t is

$$f_X(x) = \frac{2}{\Gamma\left(\frac{\nu}{2}\right)} \frac{e^{\beta(x-\mu)}}{2^{\frac{\nu+1}{2}}\sqrt{\pi\chi}} \frac{K_{\frac{\nu+1}{2}}\left(|\beta|\sqrt{\chi}\sqrt{1+\frac{(x-\mu)^2}{\chi}}\right)\left(|\beta|\sqrt{\chi}\sqrt{1+\frac{(x-\mu)^2}{\chi}}\right)^{\frac{\nu+1}{2}}}{\left(1+\frac{(x-\mu)^2}{\chi}\right)^{\frac{\nu+1}{2}}}$$

I implemented this as follows in Julia[2]:

```
using SpecialFunctions
using QuadGK

"expected value of e^X for skewed-t r.v. X with μ = 0."
function tskew_mgf1(χ,β,ν)
    if χ ≤ 0.
        throw(DomainError(χ,"requires positive χ"))
    end
    if β ≥ -0.5
        throw(DomainError(β,"requires β < -1/2"))
    end
    a = sqrt(χ*(-2β-1))
    n = ν/2
```

[2] https://julialang.org/

```
    return 2/gamma(n)*besselk(n,a)*(a/2)^n
end

"density at x of a skewed Student's-t r.v. with parameters μ, χ, β, ν"
function tskew_dens(x,μ,χ,β,ν)
    if χ ≤ 0.
        throw(DomainError(χ,"requires positive χ"))
    end
    if ν ≤ 2.
        throw(DomainError(ν,"requires ν > 2"))
    end
    n = (ν+1)/2
    a = sqrt(1+(x-μ)^2/χ)
    if abs(β) < eps()
        return a^(-2n)*gamma(n)/gamma(n-1/2)/sqrt(π*χ)
    end
    b = abs(β)*sqrt(χ)
    c = β*(x-μ)
    return 2*besselkx(n,a*b)*exp(c-a*b)*(b/(2a))^n/gamma(n-1/2)/sqrt(π*χ)
end

"""distribution function, or its complement, at x
of a skewed Student's-t r.v. with parameters μ, χ, β, ν"""
function tskew_dist(x,μ,χ,β,ν;comp=false)
    if x ≤ μ
        F = quadgk(x->tskew_dens(x,μ,χ,β,ν),-Inf,x)[1]
    else
        F = quadgk(x->tskew_dens(x,μ,χ,β,ν),x,Inf)[1]
    end
    if (x ≤ μ && comp) || (x > μ && !comp)
        return 1-F
    else
        return F
    end
end
```

Note that I am using `besselkx()` for $x \mapsto e^x K_\lambda(x)$ in the density, which limits underflow for large arguments. Also, I am setting up the integration range so that the integrand in monotonic in calculating the distribution function, which improves accuracy for large magnitude arguments.

Fitting the Parameters

Reprising the arguments from the October 30 assignment, a sequence of European-style options prices for different strike prices yields a partial description of the implied distribution function of log S_T.

$$F_{\log S_T|\mathcal{F}_t}\left(x_1^\star\right) = \frac{p_t\left(K_2, T\right) - p_t\left(K_1, T\right)}{e^{-r(T-t)}\left(K_2 - K_1\right)} \tag{1}$$

for some $x_1^\star$ between $\log K_1$ and $\log K_2$.

If we wish to fit a parametric random variable to this implied distribution, a convenient goodness-of-fit function is the Anderson-Darling metric,

$$\begin{aligned} A^2(\theta) &= n \int_{-\infty}^{\infty} \frac{(F_n(x) - F(x|\theta))^2}{F(x|\theta)\,(1 - F(x|\theta))} dF(x|\theta) \\ &= -n + n \sum_{i=1}^{n} \left(F_{i-1}^2 - F_i^2\right) \log\left(F(x_i|\theta)\right) + \left((1-F_i)^2 - (1-F_{i-1})^2\right) \log\left(1 - F(x_i|\theta)\right) \end{aligned}$$

I implemented this as

```
"goodness-of-fit based on the Anderson-Darling statistic"
function obj(θ)
    χ,β,ν = θ
    if χ ≤ 0. || β ≥ -0.5 || ν ≤ 2.
        return NaN
    end
    n = length(k_pts)
    μ₀ = -log(tskew_mgf1(χ,β,ν))
    F = map(x->tskew_dist(x,μ₀,χ,β,ν,comp=false),k_pts)
    Fc = map(x->tskew_dist(x,μ₀,χ,β,ν,comp=true),k_pts)
    terms = (imp_dist_pts[1:end-1].^2 .-imp_dist_pts[2:end].^2).*log.(F)+
        ((1 .-imp_dist_pts[2:end]).^2 .-(1 .-imp_dist_pts[1:end-1]).^2).*log.(Fc)
    return n*(sum(terms)-1)
end
```

which uses the arrays `k_pts` and `imp_dist_pts` that implement (1).

```
strike = [0.; [parse(Float64,elem[1][1].text) for elem in table_rows]];
put_close = [0.; [parse(Float64,elem[2][1].text) for elem in table_rows]];
log_money = log.(strike[2:end]/fwd);
k_pts = [log_money[1] ; (log_money[1:end-1]+log_money[2:end])/2 ; log_money[end]];
imp_dist_pts = [0.; diff(put_close)./diff(strike)/disc ; 1.];
```

The results I got using `Optim.optimize` with Nelder-Mead are in Table 1

χ	8.167×10^{-3}
β	-30.38
ν	5.791

Table 1: Skewed Student's-t parameters for the implied distribution of the logarithm of the NASDAQ-100 index on the morning of December 20, 2019, as of the close of the options market on the previous October 28.

It may be worth noting that more than 99% of the sum in the Anderson-Darling came from the half of the options with strike greater than 6925 and less than 8775. Neither wing contributes much to the fit using this metric, and this is apparent in the results.

Implied Volatilities

Calculating implied volatility from listed index options can be challenging because of the wide range of strike prices. Option prices are most sensitive to implied volatility for strike prices close to the implied forward price, and this sensitivity falls off quickly. So the demands on precision can be high.

Before we look at a possible implementation, let's go through the reduced-form Black-Scholes solution for options with European-style exercise. The typical Black-Scholes equation involves six quantities in addition to the option value: the spot price, the strike price, the term to expiration, the risk-free discount rate, the implied dividend rate, and the implied volatility rate. Note that since put/call parity means that otherwise-identical puts and calls have the same implied volatility, you only need to analyze the put or the call, not both.

These six variables in the Black-Scholes formula can be reduced to two. Consider a put for example:

$$\frac{p}{Fe^{-rT}} = e^k \Phi\left(\frac{k}{\varsigma} + \frac{\varsigma}{2}\right) - \Phi\left(\frac{k}{\varsigma} - \frac{\varsigma}{2}\right)$$

for log-moneyness $k = \log K/F$ in terms of the forward price $F = Se^{(r-\delta)T}$ and strike price K and terminal standard deviation $\varsigma = \sigma\sqrt{T}$ in terms of the implied volatility rate and term T.

For out-of-the-money options, the values are near to zero, where floating point precision is highest. But for in-the-money options, prices are significantly non-zero and precision is limited to the size of the mantissa. It would be preferable to work with values near to zero for both in-the-money and out-of-the-money options. A natural way to bring this about is to work with "standardized time-value":

$$\begin{aligned} v_{\text{mkt}}\left(\log K/F\right) &= \frac{p - e^{-rT}\max\left(0, K-F\right)}{Fe^{-rT}} = \frac{c - e^{-rT}\max\left(0, F-K\right)}{Fe^{-rT}} \\ v_{BS}(k,\varsigma) &= e^k \Phi\left(\frac{k}{\varsigma} + \frac{\varsigma}{2}\right) - \Phi\left(\frac{k}{\varsigma} - \frac{\varsigma}{2}\right) - \max\left(0, e^k - 1\right) \end{aligned} \tag{2}$$

To re-value the puts under the model, we need to do integrals of the form

$$v_{\text{model}}(k) = \int_{-\infty}^{k} \left(e^k - e^x\right) f_X\left(x|\mu = \mu_0\right)\, dx - \max\left(0, e^k - 1\right)$$

I implemented this as

```
"normalized time-values"
v_mkt = put_close[2:end]/disc/fwd.-map(k->max(0.,expm1(k)),log_money);

function v_model(k)
    χ,β,ν = θ_fit
    μ₀ = -log(tskew_mgf1(χ,β,ν))
    if k < 0.
        return quadgk(x->(exp(k)-exp(x))*tskew_dens(x,μ₀,χ,β,ν),-Inf,k)[1]
    else
        return quadgk(x->(exp(x)-exp(k))*tskew_dens(x,μ₀,χ,β,ν),k,Inf)[1]
    end
end
```

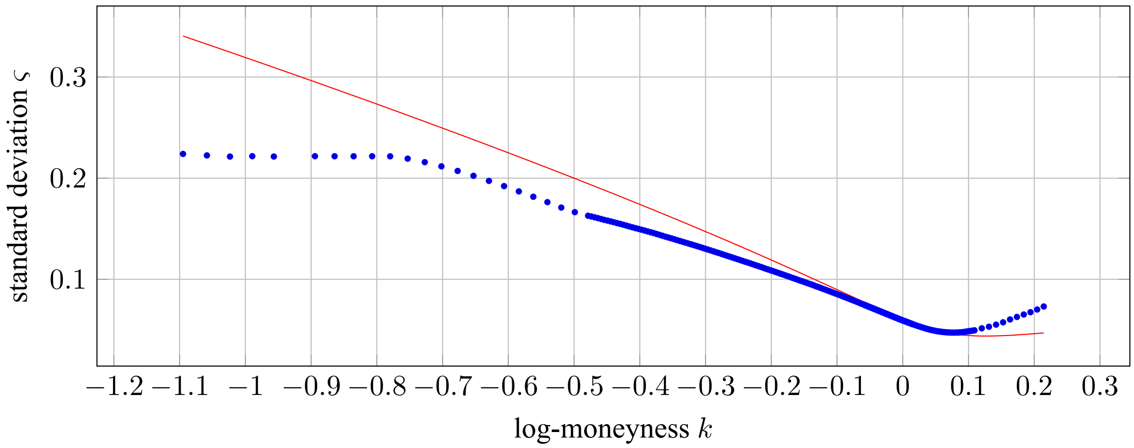

Figure 1: Implied terminal standard deviations from NASDAQ-100 options.

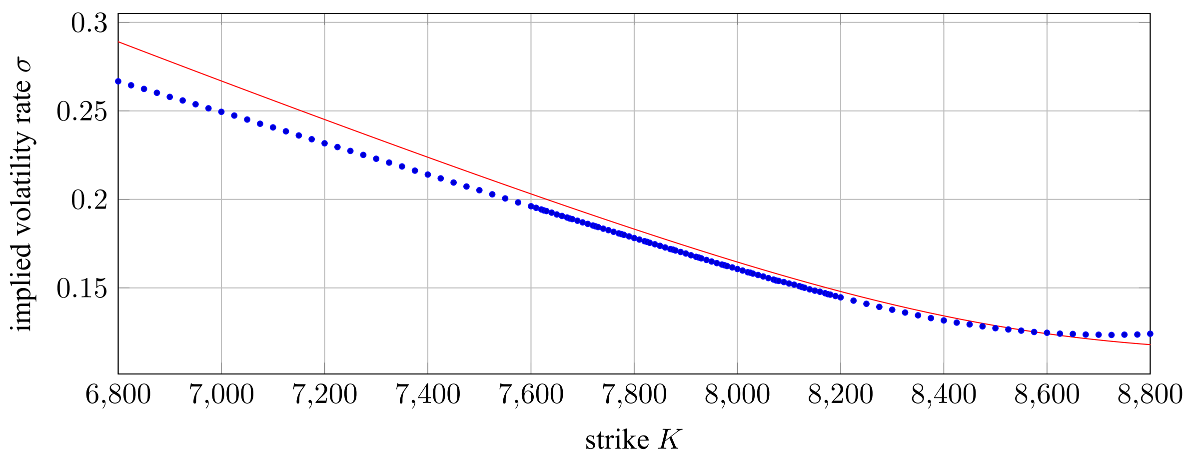

Figure 2: Near-the-money implied volatility from NASDAQ-100 options.

Since (2) cannot be inverted to give a simple expression for the function $v \mapsto \varsigma(k, v)$ we have to use numerical techniques to approximate this inversion. Existence and uniqueness is theoretically guaranteed as long as the market data is consistent with no-arbitrage, because the time-value is zero at zero volatility and strictly increasing (theoretically) without bound in volatility for any log-moneyness. Please see the Appendix for details on the implementation I used, which is a standard combination of the Newton-Raphson method and the bisection method. This seems to require at most about thirty evaluations per option to converge with our data.

The fit is not perfect, but it seems to have some accurate stylized features, including the linear wings and the negative slope at-the-money. Presumably allowing for a positive ψ would improve the fit further.

Appendix: Implied Volatility

I chose to implement a three-phase algorithm for approximating the implied volatility based on matching the logarithm of the standardized time-value of an option with the equivalent Black-Scholes value. The first phase is to bound the result with a finite interval. The second and third phase is to shrink this interval using either the Newton-Raphason algorithm or the bisection method.

The Newton-Raphson algorithm requires a function for the derivative of the standardized time value with respect to the implied standard deviation, but this is closely related to the Black-Scholes "vega" and has a simple expression.

I try to be careful to preserve precision, such as with the use of `erfcx()` for $x \mapsto e^{x^2}(1 - \text{erf}(x))$ which is more accurate for large positive arguments.

```
using SpecialFunctions

"Black-Scholes log normalized time-value"
function log_v_bs(k,ς)
    if abs(k) < eps()
        return log(erf(ς/2/sqrt(2)))
    else
        return -(k/ς-ς/2)^2/2+log(sign(k)/2*
            (erfcx(sign(k)/sqrt(2)*(k/ς-ς/2))
            -erfcx(sign(k)/sqrt(2)*(k/ς+ς/2))))
    end
end

"Black-Scholes log normalized time-value ivol sensitivity"
function log_v_bs_vega(k,ς)
    if abs(k) < eps()
        return exp(-(ς/2/sqrt(2))^2)/sqrt(2π)/erf(ς/2/sqrt(2))
    else
        return sign(k)/sqrt(π/2)/
            (erfcx(sign(k)/sqrt(2)*(k/ς-ς/2))
            -erfcx(sign(k)/sqrt(2)*(k/ς+ς/2)))
    end
end

"implied volatility from normalized time-value"
function ς_imp(k,v)
    N = 100 # maximum number of evaluations
    tol = 1.e-12
    log_v = log(v)
    ς_lo = 0.
    ς_hi = sqrt(2π)*v
    while N > 0
        if  log_v > log_v_bs(k,ς_hi)
            N -= 1
```

```
            ς_lo = ς_hi
            ς_hi *= 2.
        else
            break
        end
    end
    ς0 = (ς_lo+ς_hi)/2.
    while N > 0
        ς1 = ς0+(log_v-log_v_bs(k,ς0))/log_v_bs_vega(k,ς0)
        N -= 1
        if abs(ς1-ς0) < tol
            return ς1
        end
        diverge = ς1 > ς_hi || ς1 < ς_lo
        if ς1 > ς0
            ς_lo = ς0
        else
            ς_hi = ς0
        end
        ς0 = diverge ? (ς_lo+ς_hi)/2. : ς1
    end
    throw(ErrorException("failed to meet convergence criterion "*
        "within evaluation count limit"))
end
```