1 Introduction

The problem of calibrating option prices to market values (the "inverse problem") is non-trivial especially with complex pricing models with many parameters. A naive approach is to perform optimization by minimizing a distance between the prices provided by the market and the modeled prices by varying the input parameters. However, this can be computationally intensive. The problem is not convex and there may be a plethora of local minima. The parameter surface may have many "flat" areas leading to unstable parameter solutions.

Due to the issues arround optimization and given that practioners require efficient market calibration, we choose our pricing modeling framework in light of optimization instead of choosing the framework which may best fit the market. In a now classic paper, Carr, Madan, Geman, and Yor (2003) demonstrate that asset prices may not have a diffusion component and can be modeled as a pure jump process with infinite activity. However, Huang and Wu (2004) show that when accounting for the leverage effect, the diffusion component does have a significant impact. This is due to the empirical fact that asset returns and asset volatility are correlated. One of the only tractable ways to generate this correlation is by modeling time changes as integrated diffusions.

In our study, we find that it is very difficult to solve the inverse problem for Levy-processes of infinite activity. We therefore eschew the empirical results of CGMY and focus only on finite-activity jump diffusions a la Merton (1976). However, we do allow for a time-changed diffusion component. The models thus incorporate popular models like Heston's as a special case.

All the code is available at the following Github repo: FFTOptionPricing.

2 The model

Levy processes can be constructed with relatively simple characteristic exponents. In this paper we will focus on processes with characteristic exponents of the following form:

$$\begin{aligned}\psi(u) &= -\frac{\sigma^2 u^2}{2} + \lambda\left(1 - e^{\mu_j + \sigma_j^2/2}\right) + \lambda\left(e^{\mu_j ui - \sigma_j^2 u^2/2} - 1\right) \\ &= -\frac{\sigma^2 u^2}{2} - \psi_j(-i) + psi_j(u)\end{aligned}$$

Where $\psi_j(u) := \lambda\left(e^{\mu_j ui - \sigma_j^2 u^2/2} - 1\right)$ is the characteristic function of the jump component. $\psi(u)$ is the characteristic exponent of a Merton jump-diffusion model. Following Carr and Wu (2004), we also introduce a random time change so that the characteristic function of the normalized log asset is as follows:

$$\mathbb{E}[e^{uiX_t}] = g(-\psi(u), a, a - \sigma_v \rho u \sigma, \sigma_v, v_0)$$

Where $X_t = \log\left(\frac{S_t}{S_0}\right) - rt$, and g is the moment generating function of an integrated CIR process:

$$g(x, a, \kappa, \sigma_v, v_0) = e^{-b(t)v_0 - c(t)}$$

Where

$$b(t) = 2x\left(1 - e^{-\delta t}\right) / \left(\delta + \kappa + (\delta - \kappa)e^{-\delta t}\right)$$

$$c(t) = \left(\frac{a}{\sigma^2}\right)\left(2\log\left(1 + (\kappa - \delta)\left(1 - e^{-\delta t}\right)/2\delta\right) + \left(1 - e^{-\delta t}\right)(\kappa - \delta)\right)$$

$$\delta = \sqrt{\kappa^2 + 2x\sigma_v^2}$$

3 Calibration

Calibration has traditionally taken the following form:

$$\min_\theta \sum_k w_k \left(C_k - C(k;\theta)\right)^2$$

Where w_k is a weight, θ are the parameters describing the (risk-neutral) asset process, C_k is the observed option prices at log-strike k, and $C(k, \theta)$ is the modeled price.

As mentioned in the introduction, this problem is not trivial. See Tankov (2004) for details. Since we are dealing with Levy processes, we instead consider minimizing the following:

$$\min_\theta ||\psi(u_l;\theta) - \hat{\psi}(u_l;k)||$$

We can borrow from Carr and Madan (1999) and Belomestny and ReiÃ§ (2006) to create the estimate $\hat{\psi}$ from the observed option data:

$$\log\left(1 + iu(iu+1)\int_{-\infty}^{\infty} e^{uix} O(x)dx\right) = \psi(u - i, t)$$

Where $O(x) = C_{x+\log(S_0)+rt}/S_0 - (1 - e^x/S_0)^+$ and $x = k - \log(S_0) - rt$. Since we do not observe a continuum of option prices in the market, we use a monotonic spline to interpolate the option prices. To preserve accuracy, we use a two part spline fit. The first fit uses the realized $O_k = C_k/S_0 - \left(1 - e^{k-rT}\right)^+$ until $e^k/S_0 < 1 - \alpha$ for some α. We choose $\alpha = .05$ which seems to provide good results. The second fit is the log of normalized option prices $\log(C_k/S_0)$.

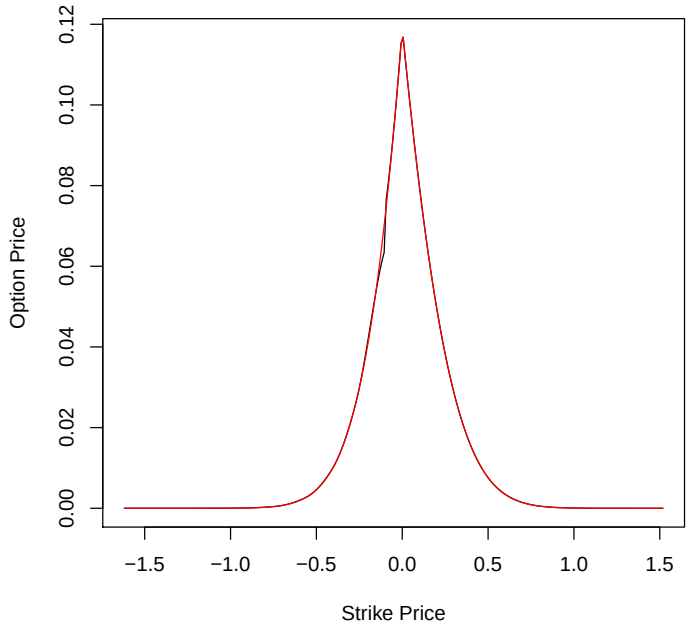
There are three sources of error in the estimate:

1. The observation error in the options themselves
2. The error in the spline approximation
3. The error in numerically integrating the observed options prices

3.1 Spline Error

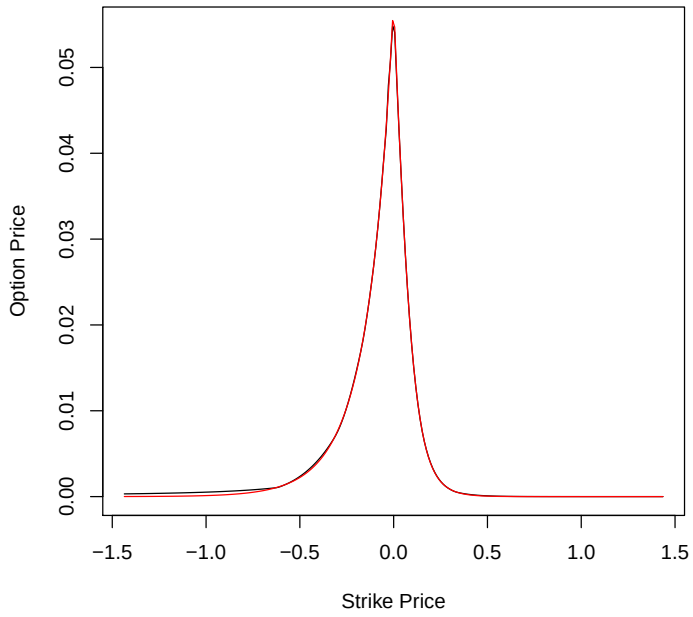
3.1.1 Black Scholes

We first test the spline error on a Black-Scholes model. The parameters chosen are $S = 10$, $r = 0$, $t = 1$, $\sigma = .3$, and the strike array $4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16$. This model shows very good alignment when choosing a spline on the log of the option price against the strike as can be seen in the following charts. Note that the black line is estimated, while the red line is actual. The red line completely covers the black line.



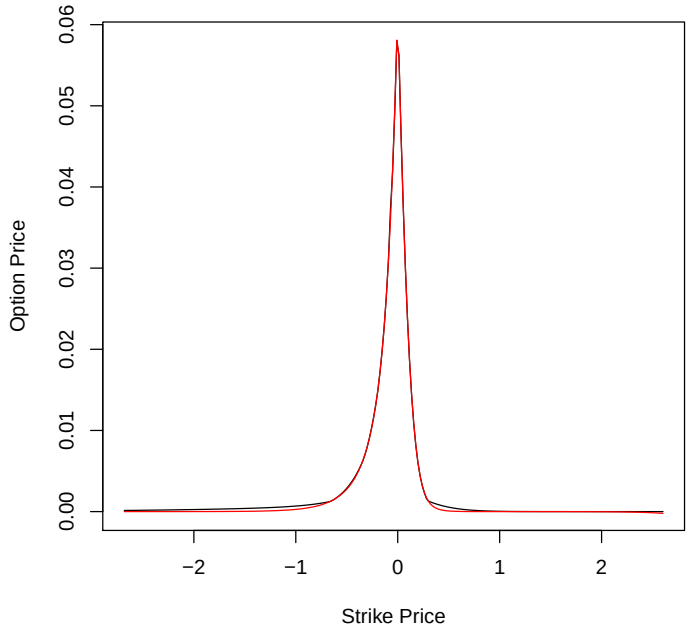
3.1.2 Heston

We then fit the spline for a Heston model. The parameters chosen are the strike array $(95, 100, 130, 150, 160, 165, 170, 175, 185, 190, 195, 200, 210, 240, 250)$.



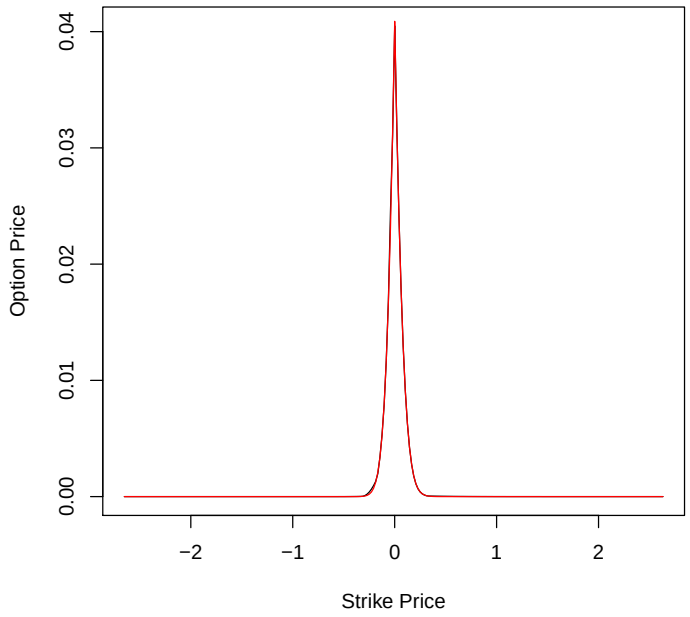
3.1.3 CGMY

We then fit the spline for a CGMY model.



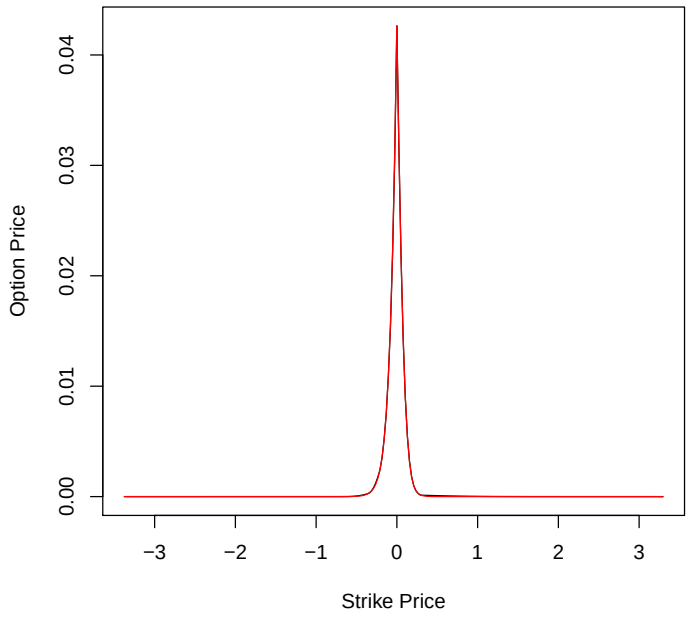
3.1.4 Merton

We then fit the spline for a Merton jump diffusion model.



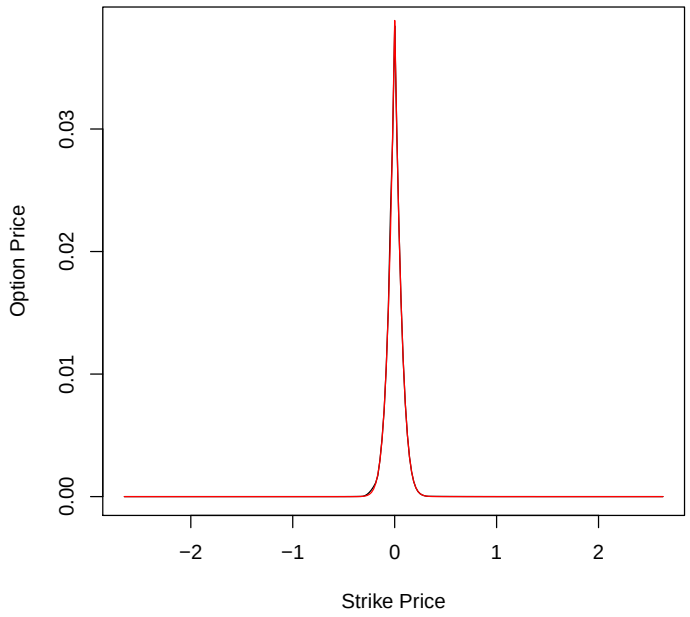
3.1.5 Merton Volatility Jumps

We then fit the spline for a Merton jump diffusion model with stochastic volatility with jumps.



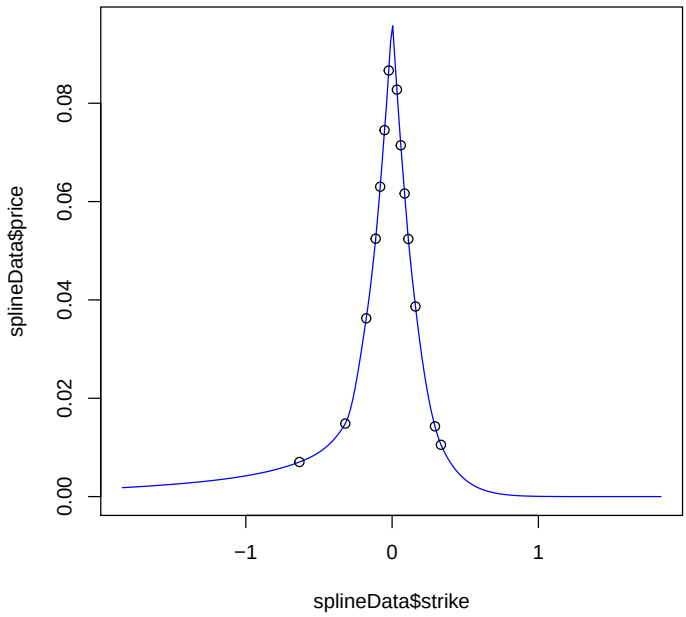
3.1.6 Merton stochastic vol

We then fit the spline for a Merton jump diffusion model with stochastic volatility



3.1.7 Actual Call prices

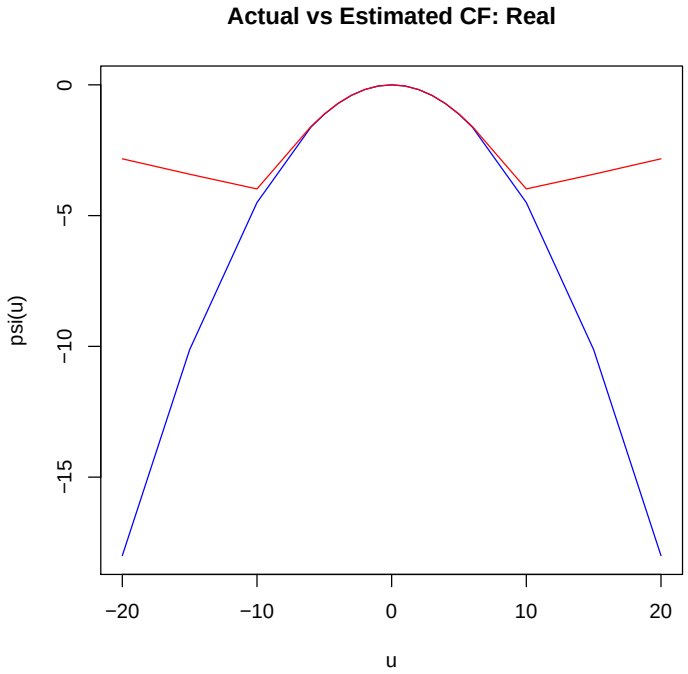
We use Apple call options on January 19 2018 with expiration one year in the future. The following chart shows the results:

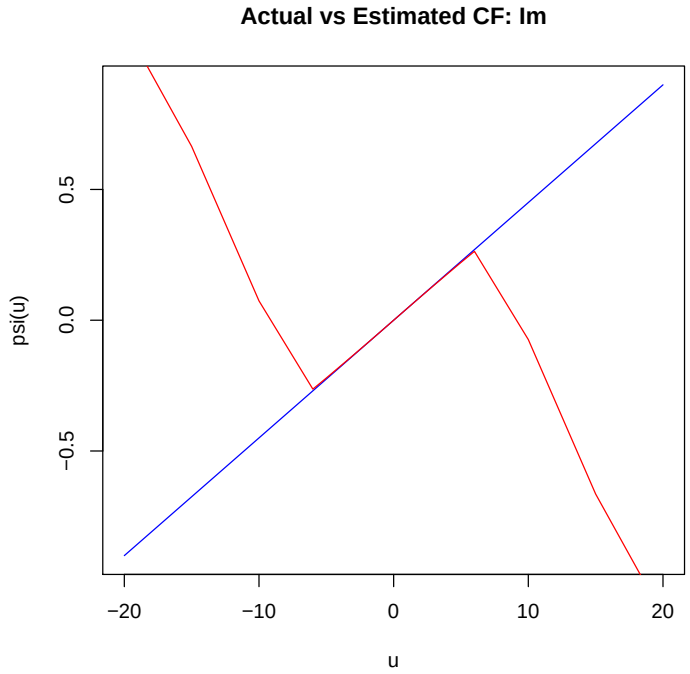


3.2 Numerical Integration Error

3.2.1 Black Scholes

The numerical integration error can be seen as follows:

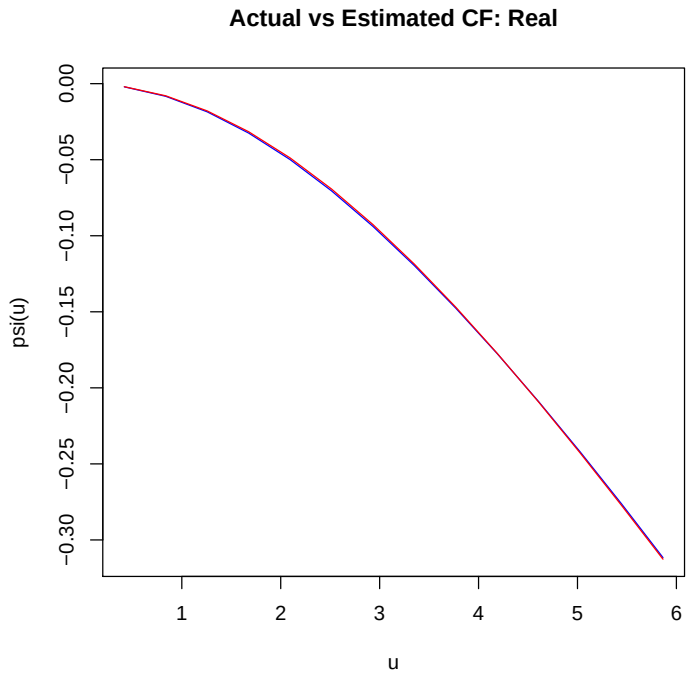

Actual vs Estimated CF: Real
psi(u)
0
−5
−10
−15
−20
−10
0
10
20
u

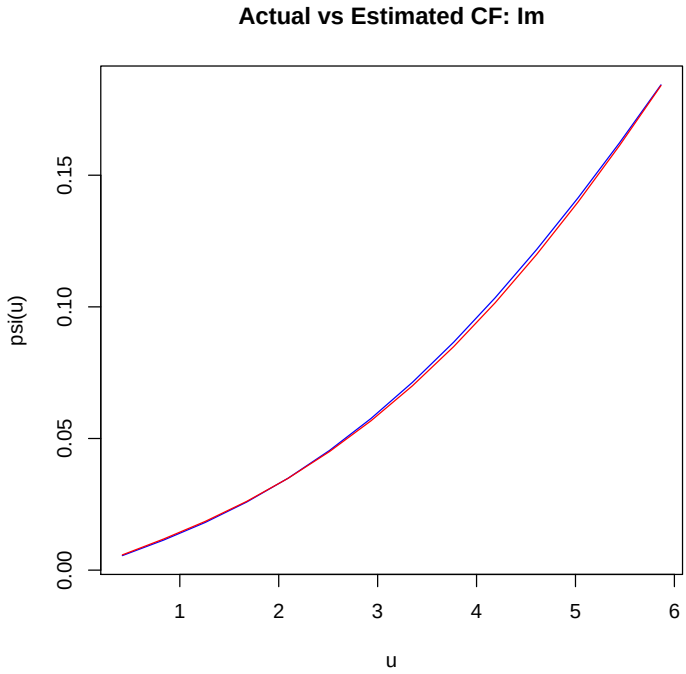


While in theory the integration should be valid over all real numbers, it appears that it is only accurate from $(-2\pi, 2\pi)$. However, since the function is even, we don't benefit from using all the domain and truncate it from $(0, 2\pi)$.

3.2.2 Heston

The following plot shows the integration comparison for a Heston model.

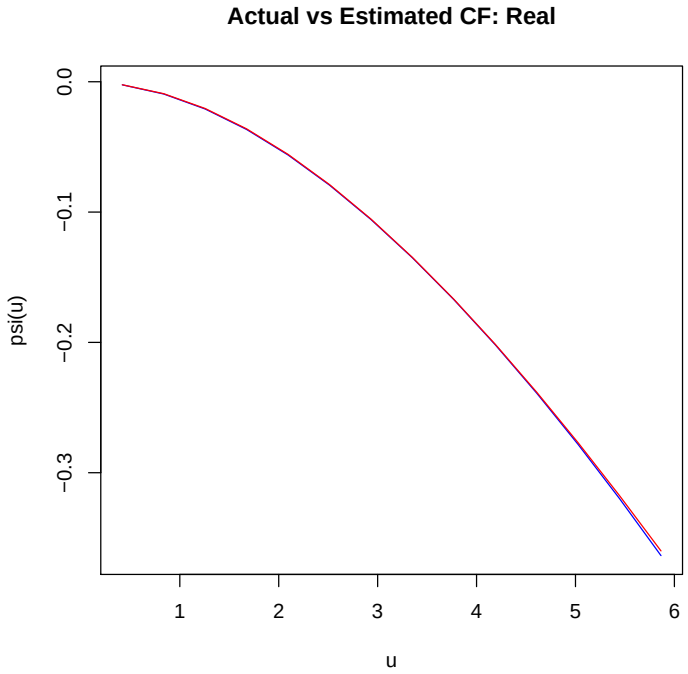

Actual vs Estimated CF: Real
psi(u)
0.00
−0.05
−0.10
−0.15
−0.20
−0.25
−0.30
1
2
3
4
5
6
u

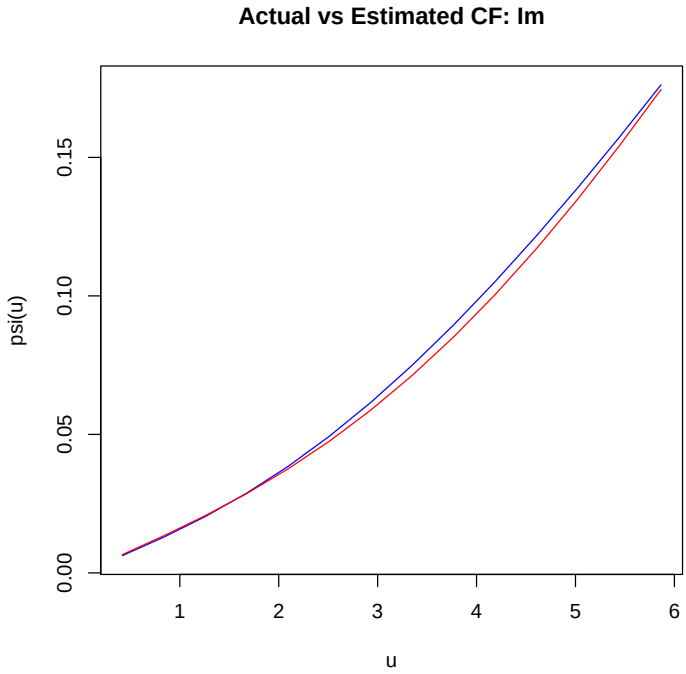


The model shows very good alignment.

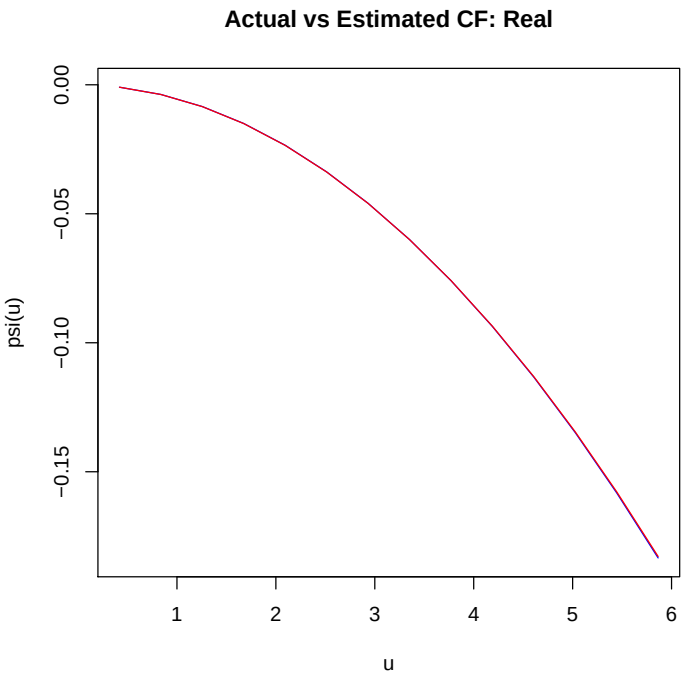
3.2.3 CGMY

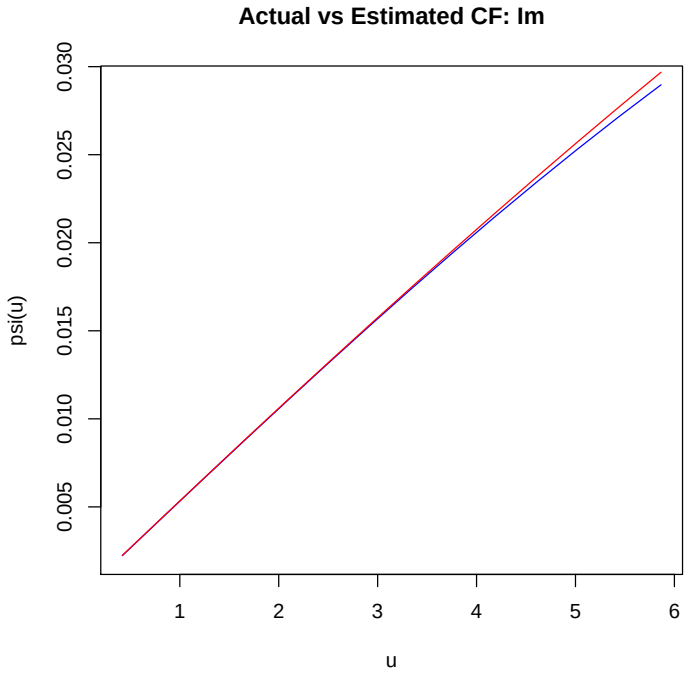
The CGMY process also shows good accuracy.


Actual vs Estimated CF: Real
psi(u)
0.0
−0.1
−0.2
−0.3
1
2
3
4
5
6
u

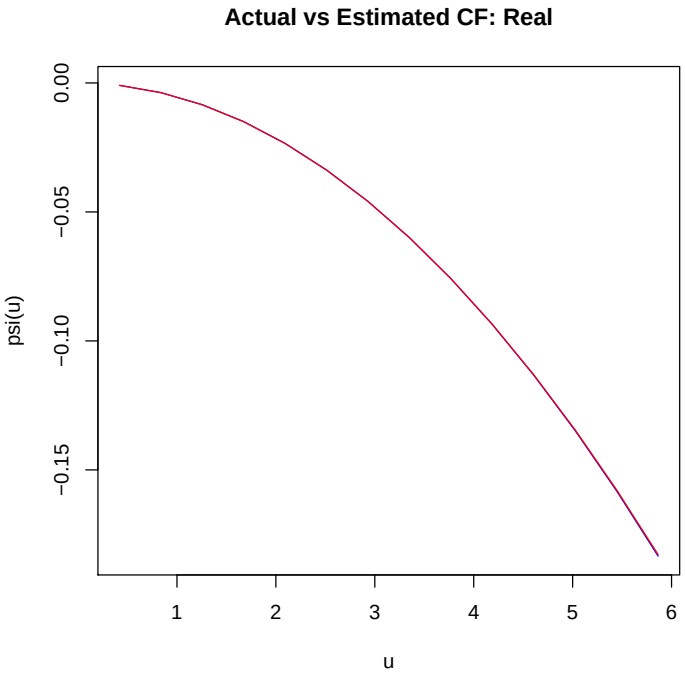

Actual vs Estimated CF: lm
psi(u)
0.00
0.05
0.10
0.15
1
2
3
4
5
6
u

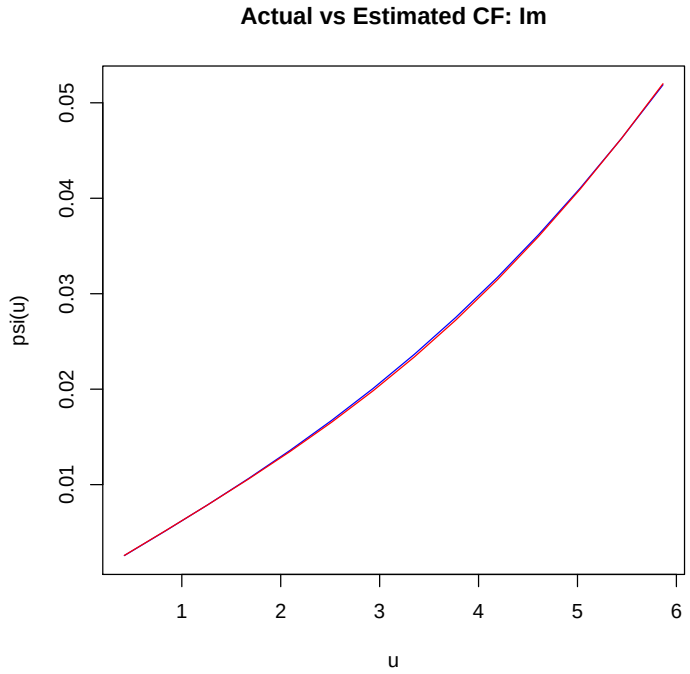
3.2.4 Merton



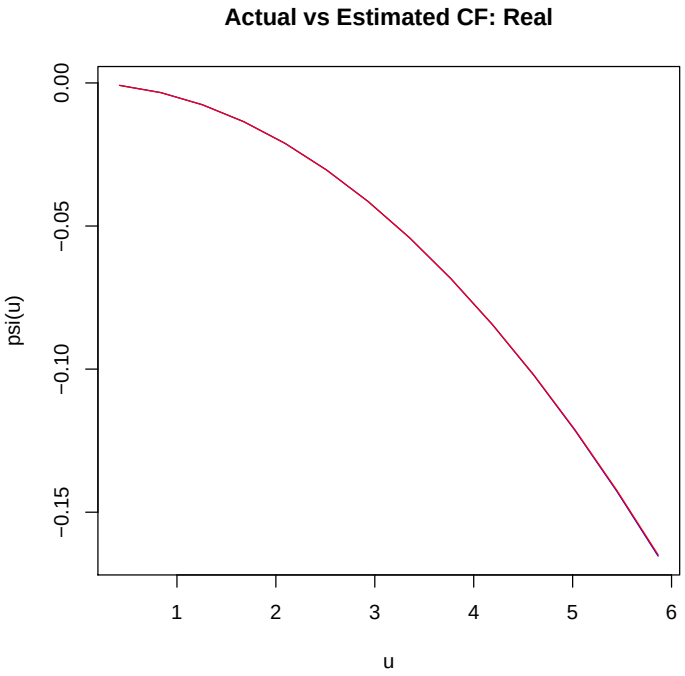

Actual vs Estimated CF: lm
psi(u)
0.005
0.010
0.015
0.020
0.025
0.030
1
2
3
4
5
6
u

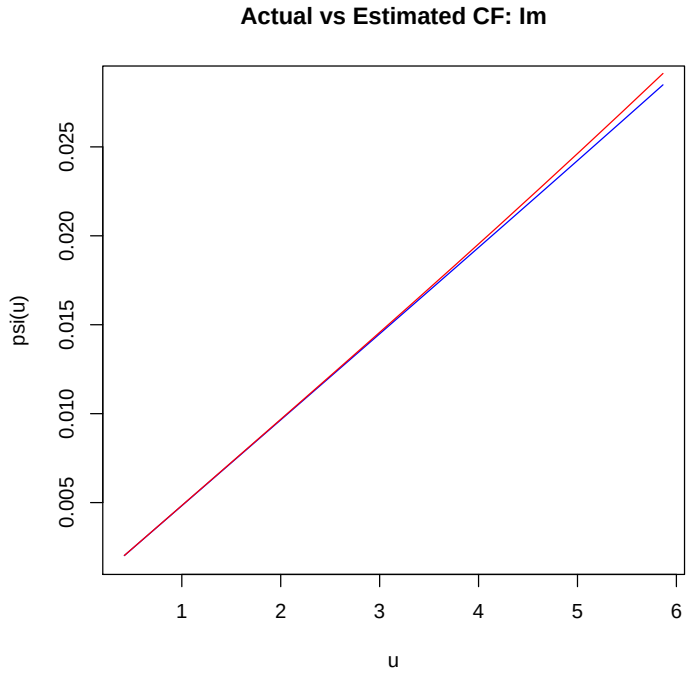
3.2.5 Merton with Time Changed Volatility with jumps




Actual vs Estimated CF: lm
psi(u)
0.01
0.02
0.03
0.04
0.05
1
2
3
4
5
6
u

3.2.6 Merton with Time Changed Volatility





3.3 Estimates

3.3.1 Black Scholes

Estimating the characteristic function at points u between negative and positive 2π, we use a Cuckoo algorithm to minimize the sum of the norms of the differences between the estimated and modeled characteristic functions. For a simple model like Black Scholes we can also use gradient descent, however for more complicated models the output has many local minima.

```
  paramater actual  optimal
1     sigma    0.3 0.299059
```

3.3.2 Heston

For more complicated models with many local minima, we use the cuckoo search. The search for a Heston model shows the following results:

```
  paramater    actual   optimal
1     sigma  0.199499  0.121767
2     speed  1.576800  0.210763
3      adaV  2.882720  2.577630
4       rho -0.571100 -0.568776
5     v0Hat  0.439698  1.964790
```

3.3.3 Merton Jump Diffusion

Again, we use the cuckoo search. The search shows the following results:

```
  paramater actual   optimal
1       sig   0.20 0.2005210
2    lambda   0.50 1.9504900
3       muJ   0.05 0.0367304
4      sigJ   0.05 0.0000000
```

3.3.4 Merton Jump Diffusion with Stochastic vol

Again, we use the cuckoo search. The search shows the following results:

```
  paramater actual   optimal
1    lambda   0.30  0.000000
2       muJ  -0.05  0.314963
3      sigJ   0.10  1.610310
4     sigma   0.10  0.108717
5        v0   0.90  1.020620
6     speed   0.30  0.000000
7      adaV   0.20  0.279430
8       rho  -0.50 -1.000000
9     delta   0.20  0.359218
```

4 Real Data

4.1 Heston

The parameters for the Apple data for Heston is the following:

```
  paramater  estimate
1     sigma 0.4085920
2     speed 1.9310200
3      adaV 1.5642100
4       rho 0.0217458
5     v0Hat 0.4409240
```

The MSE is the following:

```
  value
1  5000
```

The parameters are intuitive.

4.2 CGMY

The parameters for the Apple data for CGMY is the following:

```
  paramater  estimate
1       sig  0.215096
2         C  0.927319
3         G  3.705500
4         M  6.415600
5         Y -2.818320
```

The MSE is the following:

```
       value
1 4.0879e-05
```

4.3 CGMY with time change

```
  paramater   estimate
1       sig  0.2986990
2         C  1.8875800
3         G  4.1370400
4         M 14.9998000
5         Y -2.9178900
6     speed  1.8691400
7      adaV  0.9488360
8       rho -0.0395223
9        v0  0.0298844
```

The MSE is the following:

```
        value
1 6.01131e-05
```

The parameters are not as intuitive as for Heston.

4.4 Jump Diffusion with time change

```
  paramater    estimate
1       sig  0.32689400
2    lambda  0.00255845
3       muJ  0.29959600
4      sigJ  0.28095100
5     speed  1.96317000
6      adaV  1.87040000
7       rho -0.20535200
8        v0  0.00000000
```

The MSE is the following:

```
        value
1 8.82899e-05
```