ARCH Models in Julia

Simon A. Broda

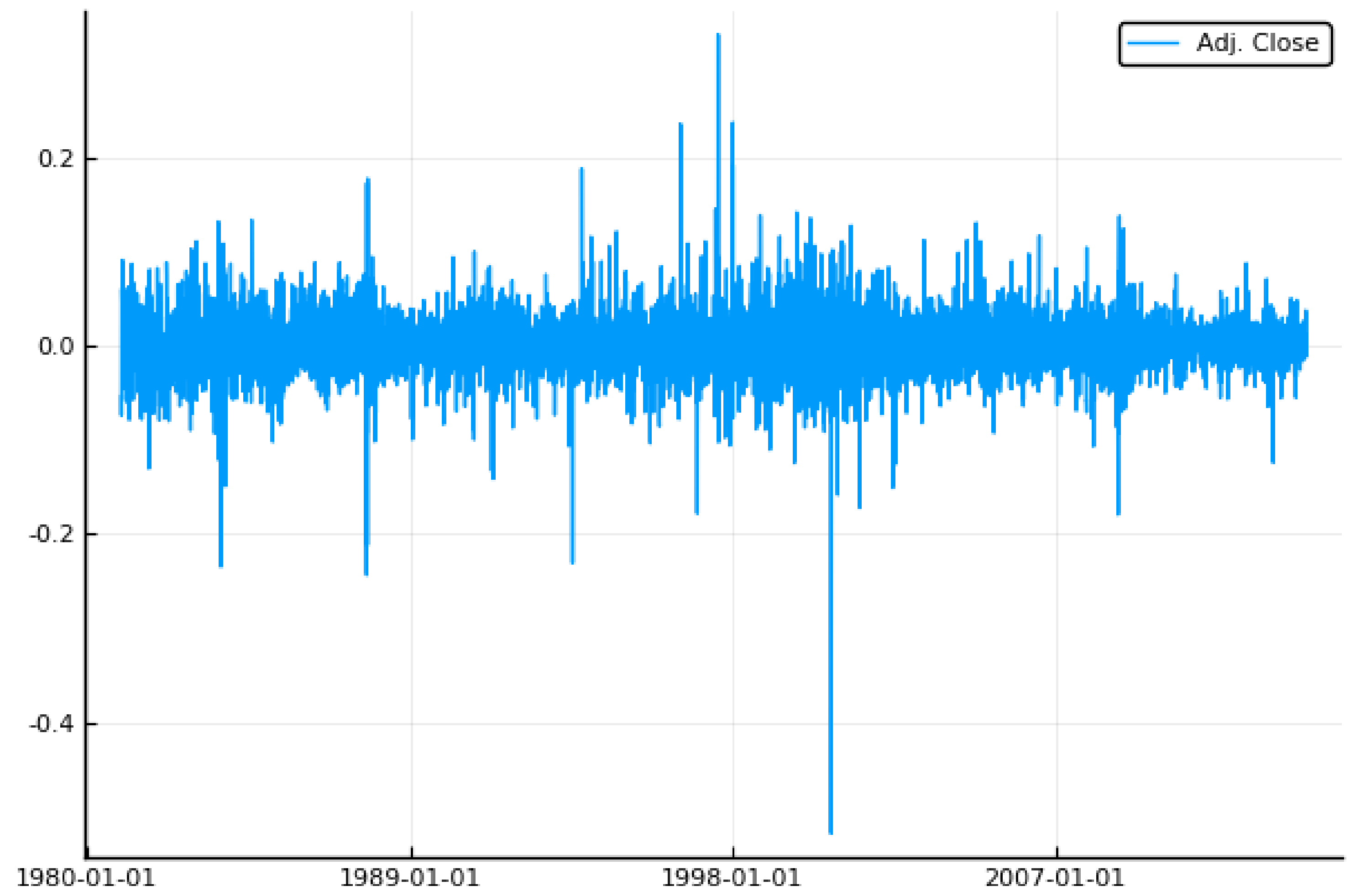
University of Zurich and University of Amsterdam
simon.broda@uzh.ch

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No. 750559).

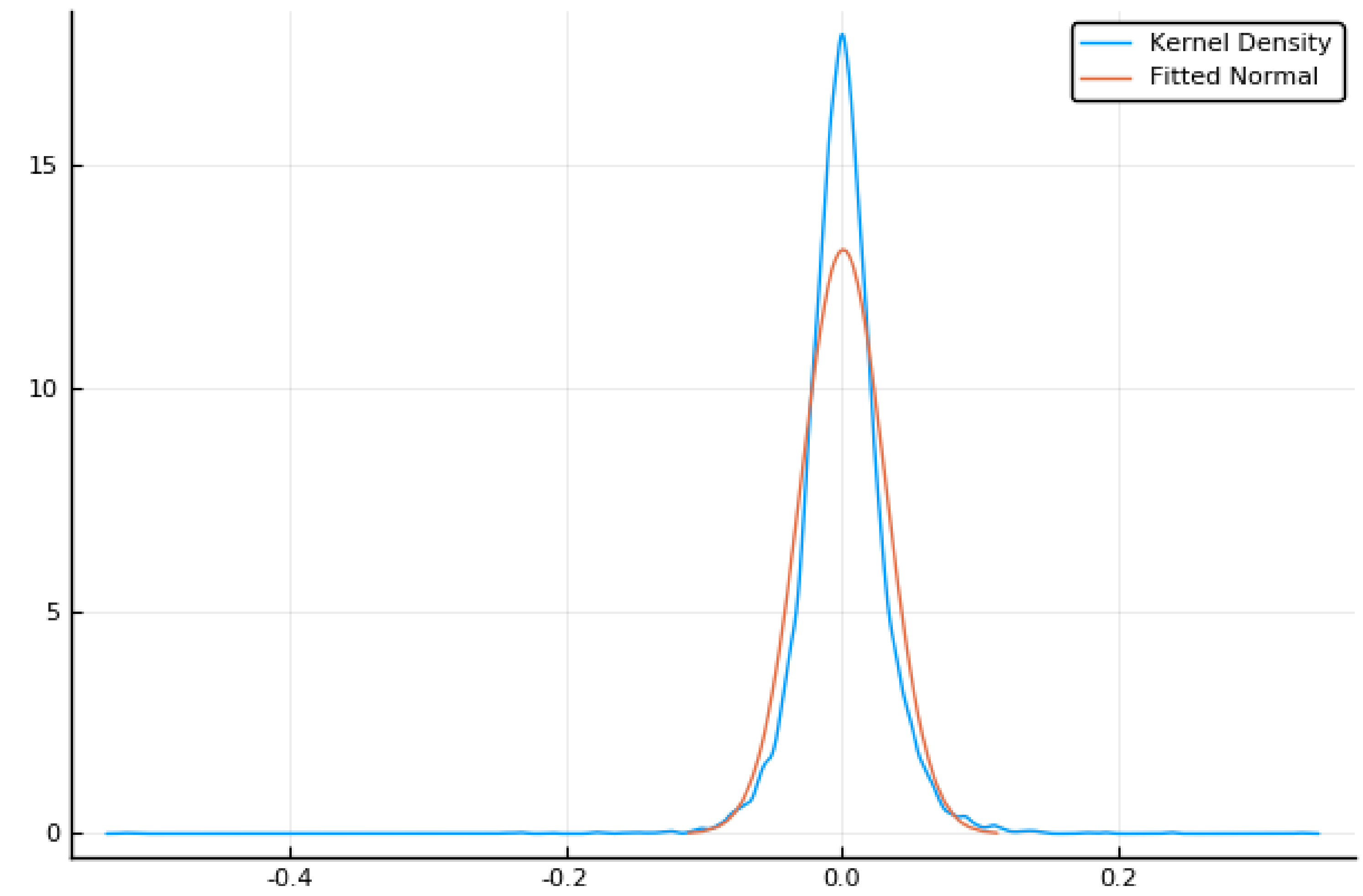
Introduction

- Daily financial returns data exhibit a number of *stylized facts*:
  - Volatility clustering
  - Non-Gaussianity, fat tails
  - Leverage effects: negative returns increase future volatility
- Other types of data (e.g., changes in interest rates) exhibit similar phenomena.
- These effects are important in many areas in finance, in particular in risk management.
- [G]ARCH ([**G**eneralized] **A**utoregressive **C**onditional **V**olatility) models are the most popoular for modelling them.

Example: volatility clustering in AAPL returns



Example: fat tails in AAPL return density



(G)ARCH Models

- Basic setup: given a sample of financial returns $\{r_t\}_{t\in\{1,\ldots,T\}}$, decompose r_t as
$$r_t = \mu_t + \sigma_t z_t, \qquad z_t \overset{i.i.d.}{\sim} (0, 1),$$
where $\mu_t \equiv \mathbb{E}[r_t \mid \mathcal{F}_{t-1}]$ and $\sigma_t^2 \equiv \mathbb{E}[(r_t - \mu_t)^2 \mid \mathcal{F}_{t-1}]$.
- Assume $\mu_t = 0$ for simplicity. Focus is on the *volatility* σ_t. G(ARCH) models make σ_t a function of *past* returns and variances. Examples:

Examples (parameter restrictions not shown)

- ARCH(q) (Engle, Ecta 1982):

$$\sigma_t = \omega + \sum_{i=1}^{q} \alpha_i r_{t-i}^2$$

- GARCH(p, q) (Bollerslev, JoE 1986)

$$\sigma_t = \omega + \sum_{i=1}^{p} \beta_i \sigma_{t-i}^2 + \sum_{i=1}^{q} \alpha_i r_{t-i}^2$$

- EGARCH(o, p, q) (Nelson, Ecta 1991)

$$\log(\sigma_t^2) = \omega + \sum_{i=1}^{o} \gamma_i z_{t-i} + \sum_{i=1}^{p} \beta_i \log(\sigma_{t-i}^2) + \sum_{i=1}^{q} \alpha_i (|z_t| - \mathbb{E}|z_t|)$$

Estimation

- G(ARCH) models are usually estimated by maximum likelihood: with f denoting the density of z_t,

$$\max \prod_t \frac{1}{\sigma_t} f(r_t/\sigma_t).$$

- Recursive nature of σ_t means the computation cannot be "vectorized" $\Rightarrow$ loops.
- Julia is very well suited for this. Matlab (and the `rugarch` package for Python) have to implement the likelihood in C.

The ARCH Package

- ARCH.jl is not registered yet; available at https://github.com/s-broda/ARCH.jl
- 0.6 only so far; 0.7 support coming soon.
- supported so far: simulation and estimation for ARCH, GARCH, and EGARCH models of arbitrary orders, with Gaussian and Student's t errors.
- Designed to be easily extensible with new models, distributions.
- Volatility specifications subtype `VolatilitySpec`. Parametrized on (o, p, q) to facilitate loop unrolling.
- Simulation and estimation return instances of `ARCHModel`, which subtypes `StatisticalModel` from `StatsBase`.
- Standard errors obtained by AD via `ForwardDiff.jl`.

Usage

In [2]:

```
using Suppressor #silence some method overwrite warnings
@suppress using ARCH
srand(1); T=10^4 #sample size
volaspec = GARCH{1, 1}([1., .9, .05]) #[omega, beta, alpha]
am = simulate(volaspec, T; dist=StdTDist(3.)) #returns ARCHModel
fit(GARCH{1, 1}, am.data; dist=StdTDist) #returns ARCHModel
```

Out[2]:

```
GARCH{1,1} model with Student's t errors, T=10000.


Mean equation parameters:

       Estimate Std.Error  z value Pr(>|z|)
μ    0.00312101 0.0282643 0.110422   0.9121

Volatility parameters:

      Estimate  Std.Error z value Pr(>|z|)
ω      1.01605   0.160286 6.33896    <1e-9
β₁    0.898115  0.0120845 74.3195   <1e-99
α₁   0.0548009 0.00750701 7.29996   <1e-12

Distribution parameters:

     Estimate Std.Error z value Pr(>|z|)
ν     2.93798  0.095268 30.8391   <1e-99
```

In [3]:
```
#select an EGARCH model without intercept by minimizing AIC; o, p, q < 3
selectmodel(EGARCH, am.data; meanspec=NoIntercept, criterion=aic, maxlags=2, dist=StdTDist)
```

Out[3]:
```
EGARCH{1,1,1} model with Student's t errors, T=10000.


Volatility parameters:

       Estimate  Std.Error  z value Pr(>|z|)
ω      0.155388  0.0234126  6.63692   <1e-10
γ₁   0.00550377 0.00945916 0.581846   0.5607
β₁     0.954636 0.00730722  130.643   <1e-99
α₁     0.145216  0.0149484  9.71446   <1e-21

Distribution parameters:

     Estimate Std.Error z value Pr(>|z|)
ν     2.92954 0.0945454 30.9856   <1e-99
```

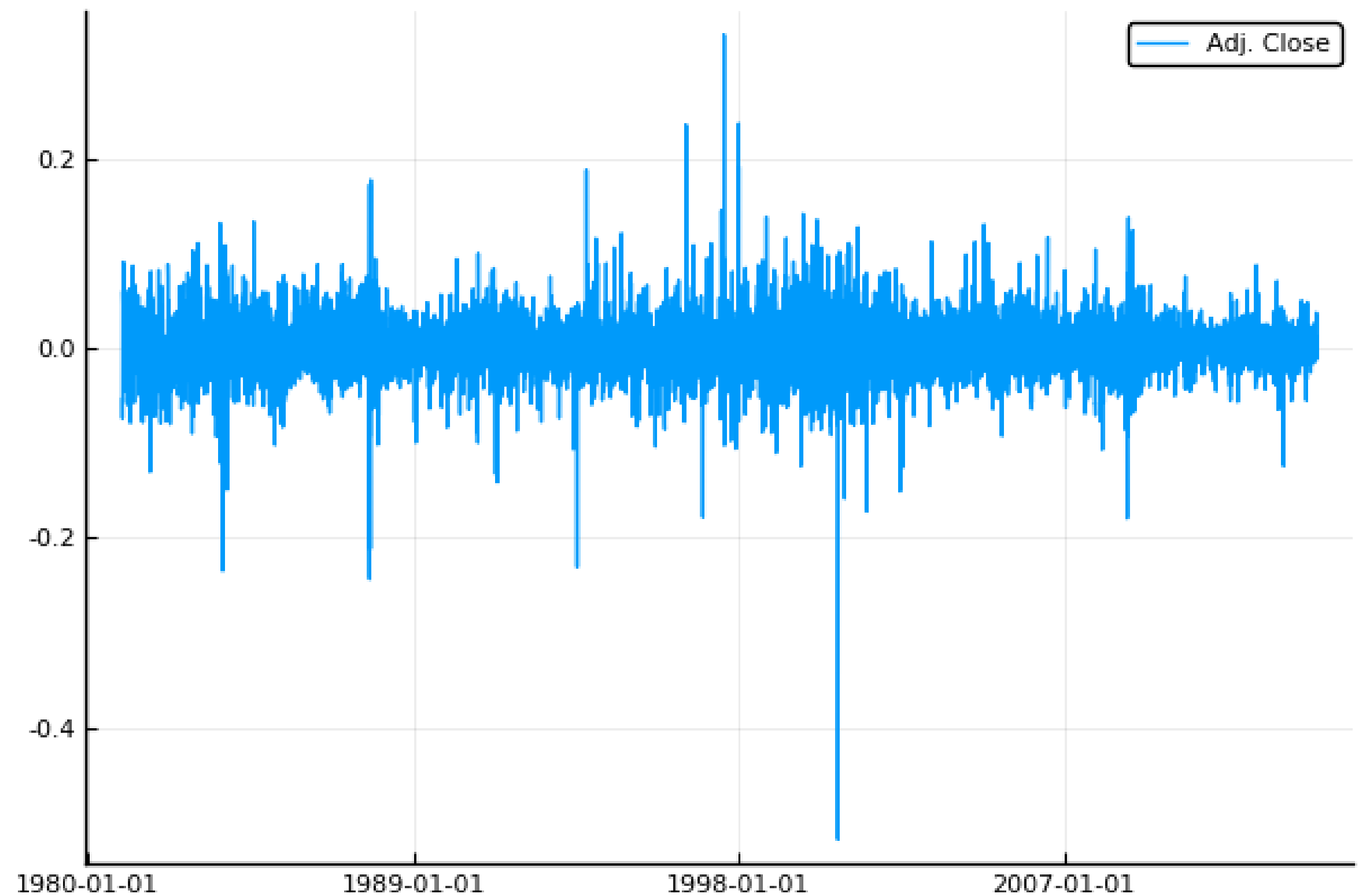
Benchmarks

- Bollerslev and Ghysels (JBES 1996) data is de facto standard in comparing implementations of GARCH models.
- Data consist of daily German mark/British pound exchange rates (1974 observations).

In [4]:

```
@static if !isfile("DMGBP.txt")
    using HTTP
    open("DMGBP.txt", "w") do io
        HTTP.get("http://people.stern.nyu.edu/wgreene/Text/Edition7/TableF20-1.txt", response_stream=io)
    end
end
r = convert.(Float64, readcsv("DMGBP.txt")[2:end]);
@static if !isfile("DMGBP.png")
    using Plots
    plot(r)
    savefig("DMGBP")
end
```

Bollerslev and Ghysels data



GARCH

- Fitting in Julia:

In [5]:
```
using BenchmarkTools
@btime fit(GARCH{1, 1}, $r, meanspec=NoIntercept) #Matlab doesn't use an intercept
```

Out[5]:
```
GARCH{1,1} model with Gaussian errors, T=1974.


Volatility parameters:

      Estimate  Std.Error z value Pr(>|z|)
ω    0.0110992 0.00665741 1.66719   0.0955
β₁    0.804287  0.0744193 10.8075   <1e-26
α₁    0.149875  0.0543414 2.75803   0.0058


  15.361 ms (2339 allocations: 113.66 KiB)
```

- Now Matlab:

In [6]:
```
using MATLAB
```

In [7]:
```
#run this cell a few times to give Matlab a fair chance
mat"tic; estimate(garch(1, 1), $r); toc; 0";
```

```
    GARCH(1,1) Conditional Variance Model (Gaussian Distribution):

                  Value       StandardError    TStatistic      PValue  
                ________      _____________    __________    __________

    Constant    0.010868        0.0012972        8.3779      5.3896e-17
    GARCH{1}     0.80452         0.016038        50.162               0
    ARCH{1}      0.15433         0.013852        11.141      7.9448e-29

Elapsed time is 0.100213 seconds.
```

- ARCH.jl is faster by a factor of about 6-10, depending on the machine.
- Estimates are quite similar, but standard errors and t-statistics differ.
- So which standard errors are correct? Let's compare with the results from Brooks et. al. (Int. J. Fcst. 2001).
- They use a model with intercept, so let’s re-estimate in Julia (not sure how to include an intercept in Matlab):

In [8]:
```
@btime fit(GARCH{1, 1}, $r)
```

Out[8]:
```
GARCH{1,1} model with Gaussian errors, T=1974.


Mean equation parameters:

        Estimate  Std.Error   z value Pr(>|z|)
μ    -0.00626931 0.00923712 -0.678709   0.4973

Volatility parameters:

      Estimate  Std.Error z value Pr(>|z|)
ω    0.0109834 0.00657626 1.67016   0.0949
β₁    0.805808   0.073853  10.911   <1e-26
α₁      0.1487  0.0540332   2.752   0.0059


  20.925 ms (2786 allocations: 146.86 KiB)
```

- Brooks et. al. give the estimates (t**-stats**) $\mu = -0.00619(\mathbf{-0.67})$, $\omega = 0.0108(\mathbf{1.66})$, $\beta_1 = 0.806(\mathbf{11.11})$, $\alpha_1 = 0.153(\mathbf{2.86})$.
- We are pretty close. Difference is due to the way presample values are computed. We can match Brooks et. al. exactly if we use the historical instead of the unconditional variance.

EGARCH

- Julia:

In [9]:
```
@btime fit(EGARCH{1, 1, 1}, $r, meanspec=NoIntercept)
```

Out[9]:
```
EGARCH{1,1,1} model with Gaussian errors, T=1974.


Volatility parameters:

      Estimate Std.Error  z value Pr(>|z|)
ω    -0.133845 0.0504588 -2.65255   0.0080
γ₁   -0.032647 0.0251908 -1.29599   0.1950
β₁    0.909471 0.0325792  27.9157   <1e-99
α₁     0.33232 0.0694224  4.78693    <1e-5


  33.674 ms (2986 allocations: 161.73 KiB)
```

- Matlab:

```
In [10]: mat"tic; estimate(egarch(1, 1), $r); toc; 0" #Matlab sets o=q
```

```
Out[10]: 0.0
```

```
    EGARCH(1,1) Conditional Variance Model (Gaussian Distribution):

                      Value       StandardError    TStatistic      PValue
                    _________     _____________    __________    __________

    Constant          -0.1283       0.015788         -8.1267     4.4118e-16
    GARCH{1}          0.91186      0.0084535          107.87              0
    ARCH{1}           0.33317       0.021769          15.305     7.1324e-53
    Leverage{1}     -0.032252       0.012564          -2.567       0.010258

Elapsed time is 0.150402 seconds.
```

- Brooks et. al. give no benchmark results. But again, Julia is faster by a factor of about 6-10.

TODO

- 0.7 compatibility
- docs
- forecasting
- more models, distributions
- Value at Risk
- backtesting
- MGARCH

References

- Bollerslev, T (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics* **31**, 307–327.
- Bollerslev, T. & Ghysels, E. (1996). Periodic Autoregressive Conditional Heteroscedasticity. *Journal of Business & Economic Statistics* **14**, 139-151. https://doi.org/10.1080/07350015.1996.10524640.
- Brooks, C., Burke, S. P., & Persand, G. (2001). Benchmarks and the accuracy of GARCH model estimation. *International Journal of Forecasting* **17**, 45-56. https://doi.org/10.1016/S0169-2070(00)00070-4.
- Engle, R. F. (1982). Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation. *Econometrica* **50**, 987-1007. https://doi.org/10.2307/1912773.
- Nelson, D.B. (1991). Conditional Heteroskedasticity in Asset Returns: A New Approach. *Econometrica* **59**, 347--370. https://doi.org/10.2307/2938260.