Bayesian Inference and Volatility Modeling Using Stan

Part II: The Bayesian Workflow

Michael Weylandt ∪ Luis Damiano

R/Finance 2018 | 2018-06-01

Motivating example (1)

We compute the daily log-returns of SPY from 1993-01-01 to 2018-05-18...

```r
library(doParallel)
library(quantmod)
library(moments)
library(rstan)
library(xts)
source("R/cache.R")
source("R/plots.R")
source("R/models.R")

rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())

p <- getSymbols(
  "SPY",
  src  = "yahoo",
  from = "1993-01-01",
  to   = "2018-05-18",
  auto.assign = FALSE
)

y <- na.omit(ROC(log(Cl(p)), n = 1, type = "continuous"))
```

Motivating example (2)

. . . and we analyze a deterministic volatility model from the Bayesian point of view.

```r
fit <- stan(
    file    = "stan/garch11.stan",
    data    = list(
      T      = length(y),     # Length of the series
      H      = 1,             # Produce H-step-ahead forecast
      y      = as.numeric(y), # The observed series (daily log-returns)
      sigma1 = sd(y)          # The conditional volatility at t = 1
    ),
    chains  = 4,
    iter    = 500,
    warmup  = 250,
    seed    = 9000,
    verbose = FALSE
  )

shat  <- apply(extract(fit, pars = "sigma")[[1]], 2, median)
spred <- extract(fit, pars = "spred")[[1]]
ypred <- extract(fit, pars = "ypred")[[1]]
sfore <- extract(fit, pars = "sfore")[[1]]
yfore <- extract(fit, pars = "yfore")[[1]]
```

Bayesian Workflow

Scientific problem

We would like to measure and forecast daily volatility in stock prices.

Substantive knowledge

What do we know about returns and volatility? (Cont 2001)

- Unconditional heavy tails
- Gain/loss asymmetry (except for foreign exchange rates)
- Shape of distribution changes with time scale
- Volatility clusters
- Conditional heavy tails (after correcting for volatility clustering)
- Slow decay of autocorrelation in absolute/squared returns
- Leverage effect
- Others

Use the model and/or priors to formally account for your substantive knowledge

Exploratory Data Analysis

A method to build a network of increasingly complex models that capture features and heterogeneities present in the data (Gelman 2004).

What are the possible sources of heterogeneity?

- Non-linearity
- Time-varying properties/relationships (coefficients)
- Hierarchies
- Clusters
- Latent variables (ex. latent states)

Model (1)

GARCH model introduced by Engle (1982) and Bollerslev (1986).

$$
\begin{aligned}
y_t &\sim \mathcal{N}(0, \sigma_t^2) \\
\sigma_t^2 &= \alpha_0 + \alpha_1 y_{t-1}^2 + \beta\sigma_{t-1}^2
\end{aligned}
$$

where σ_t^2 is the conditional variance at time t, y_t is the (de-filtered) log-return at time t.

Parameter constraints: $\alpha_0 > 0$, $\alpha_1, \beta \geq 0$, $\alpha_1 + \beta < 1$ so that the unconditional variance is finite and positive and the conditional variance is positive.

Model (2)

Interpretation (Alexander 2008):

- **Error parameter**: α_1 measures the reaction of conditional volatility to market shocks. The larger the coefficient, the more sensitive to unexpected movements in prices.
- **Lag parameter**: β measures the persistence in conditional volatility. The larger the coefficient, the longer it takes for volatility to die out.
- **Rate of convergence**: $\alpha_1 + \beta$ measures how quickly conditional volatility converges to long-term average. The closer to one, the slower the convergence.

Generative model

The **predictive prior distribution** $p(y)$ is the distribution of the unknown but observable y **before** considering our sample.

$$p(y) = \int p(y, \theta) d\theta = \int p(\theta) p(y|\theta) d\theta$$

- **Generative models**: Bayesian models with proper priors.
- **Goal**: To understand the model structure before making the measurements.
- **Methodology**: Visualize simulations from the prior marginal distribution and assess consistency between chosen priors and domain knowledge.
- **Recommendations** (Gabry et al. 2017):
  - At least some mass around extreme but plausible data sets (ex. extreme log-returns).
  - No mass on completely implausible values (ex. negative prices or volume).
  - Beware commonly recommended "vague priors" may **not** make sense in the application.

Generative model - An example

```r
fit <- stan(
    file    = "stan/garch11gen.stan",
    data    = list(
      T      = 252 * 10,       # Simulate a 10-year long sample
      sigma1 = 0.0025          # Arbitrary value
    ),
    chains  = 4,
    iter    = 500,
    warmup  = 250,
    seed    = 9000,
    verbose = FALSE
  )

spred <- extract(fit, pars = "spred")[[1]]
ypred <- extract(fit, pars = "ypred")[[1]]
```

Generative model - Data and parameters

```
data {
  int<lower=0> T;
  real<lower=0> sigma1;
}

parameters {
  real<lower=0> alpha0;
  real<lower=0, upper=1> alpha1;
  real<lower=0, upper=(1-alpha1)> beta1;
}
```

Note: there is no sample vector in the `data` block! We have **not** seen the dataset as of now.

Generative model - Model and priors

```
model {
  // Priors
  alpha0 ~ normal(0, 0.5) T[0, ]; // close to zero and small
  alpha1 ~ beta(0.50, 1.50);      // slightly more likely to be close to zero
  beta1  ~ beta(2.50, 0.80);      // slightly more likely to be close to one
}
```

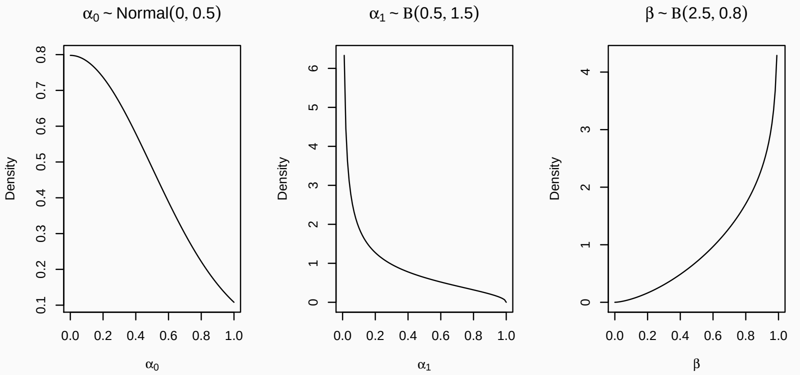
From previous experience, we expect that:

- $\alpha_0 > 0$ is very close to zero.
- $\alpha_1 \geq 0$ is close to zero, with values above 0.1 being "relatively large".
- $\beta \geq 0$ is close to one, with commonly-seen values in the range 0.70/0.99.

Note:

- There are no calls to sampling statements (log-likelihood) in the `model` block.
- Independent priors for each parameter do not account for constraints. Stan will take care of that for us :).

Generative model - Model and priors

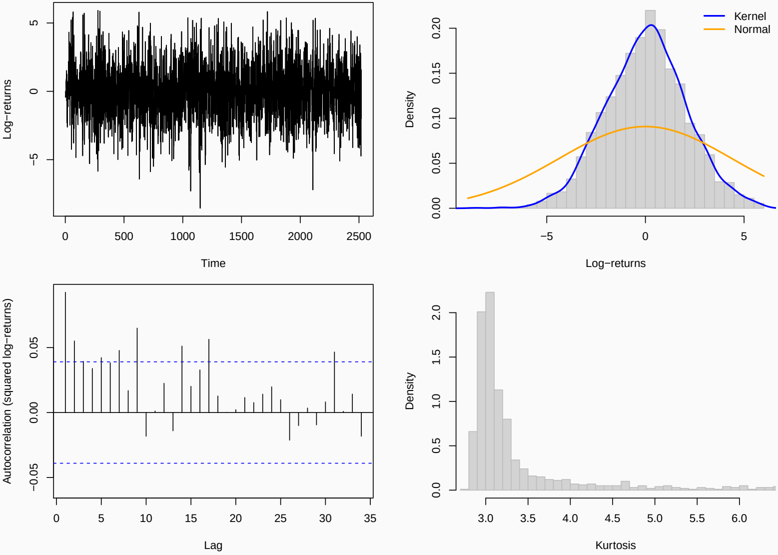


Weakly informative priors: although we use information from previous empirical studies to **regularize and stabilize** the density, we are conservative.

Generative model - An example

```
generated quantities {
  real<lower=0> spred[T];
  real ypred[T];

  spred[1] = sigma1;
  ypred[1] = normal_rng(0, sigma1);
  for (t in 2:T) {
    spred[t] = sqrt(
      alpha0
      + alpha1 * pow(ypred[t-1], 2)
      + beta1 * pow(spred[t-1], 2)
    );
    ypred[t] = normal_rng(0, spred[t]);
  }
}
```

Generative model - An example



Software validation

Before we analyze any estimates, we need to confirm that our software works as intended (Cook, Gelman, and Rubin 2006). For $i \in 1, \dots, N$ replications:

1. Draw one sample of the parameter vector $\theta^{(i)}$ from the prior distributions $p(\theta)$.
2. Draw one sample of the observation vector $y^{(i)}$ from the sampling distribution $p(y|\theta^{(i)})$.
3. Use Stan to estimate the model parameters.
4. Can the software recover the true parameters systematically?
    - Are the estimates reasonable (for example, do they take impossible values)?
    - Is there any bias (systematic error in estimation)?
    - Is the true value adequately covered by the posterior intervals?

Software validation - Step 1

$$\theta^{(i)} \sim p(\theta)$$

```
simParams <- function() {
  alpha0 = abs(rnorm(n = 1, mean = 0, sd = 0.5)) # Half-normal
  alpha1 = rbeta(n = 1, shape1 = 0.5, shape2 = 1.5)
  beta1  = rbeta(n = 1, shape1 = 2.5, shape2 = 0.8)
  sigma1 = rexp(n = 1) / 100
  if (alpha1 + beta1 >= 1) {
    simParams()
  } else {
    c(sigma1, alpha0, alpha1, beta1)
  }
}
```

Software validation - Step 2

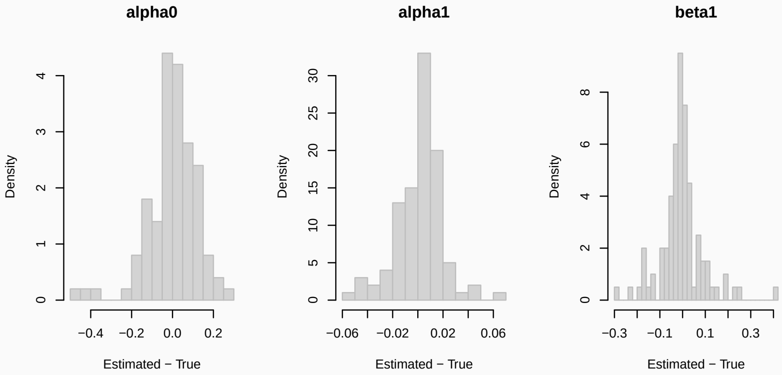
$$y^{(i)} \sim p(y|\theta^{(i)})$$

```
simGARCH11 <- function(T, sigma1, alpha0, alpha1, beta1) {
  if (sigma1 <= 0) { stop("Please use a positive sigma for the first step.")}
  if (alpha0 <= 0) { stop("Alpha0 must be greater than zero.") }
  if (min(alpha1, beta1) < 0) { stop("Alpha1 and Beta1 can't be negative.") }
  if (alpha1 + beta1 >= 1) { stop("Alpha1 + Beta1 cannot be equal to or exceed one.") }

  y <- vector(mode = "numeric", length = T)
  s <- vector(mode = "numeric", length = T)

  s[1] <- sigma1
  y[1] <- rnorm(n = 1, mean = 0, sd = s[1])
  for (t in 2:T) {
    s[t] <- sqrt(alpha0 + alpha1 * y[t - 1]^2 + beta1 * s[t - 1]^2)
    y[t] <- rnorm(n = 1, mean = 0, sd = s[t])
  }

  return(y)
}
```

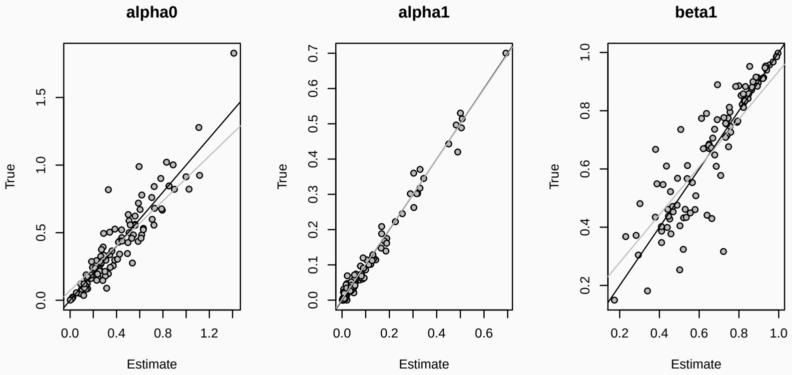
Software validation - Step 3

```r
N <- 100
set.seed(9000)
fitfakes <- lapply(1:N, function(i) {
  params <- simParams()
  yfake  <- simGARCH11(252 * 10, params[1], params[2], params[3], params[4])
  fit    <- stan(
      file    = "stan/garch11.stan",
      data    = list(
        T      = length(yfake),     # Length of the series
        H      = 1,                 # Produce H-step-ahead forecast
        y      = as.numeric(yfake), # The observed series (daily log-returns)
        sigma1 = params[1]          # The conditional volatility at t = 1
      ),
      chains  = 4,
      iter    = 500,
      warmup  = 250,
      seed    = 9000,
      verbose = FALSE
  )

  list(
    true = params,
    estimates = extract(fit, pars = c("alpha0", "alpha1", "beta1"))
  )
})
```

Software validation - Step 4



Software validation - Step 4



Software validation - Step 4

Other diagnostics:

- The 95% posterior intervals for the parameters α_0, α_1, and β contain the true value in the 89, 97, 93 percent of samples respectively.
- All values for the estimated parameters are within the restricted parameter space.
- MCMC chains mix well[1].

[1] See, for example Bayesplot

Model diagnostics

After constructing a probability model and computing the posterior distribution, we assess the fit of the model to **the data and our substantive knowledge**.

- Have we included **all** our knowledge about the problem?
- What aspects of reality are not captured by the model?
- Suspects: priors, likelihood, model structure, explanatory variables.

Model diagnostics

- **Sensitivity analysis**: how much does posterior inference change when other reasonable priors and/or models are used?
- Judge by **practical implication**: there is **no true model**.
- Does inference **make sense?**
  - Not all knowledge is included formally in the model.
  - Use the substantive leftovers to analyze the results.
- **External validation**: predict **future** data and compare with future observations.

Posterior predictive checks

If the model fits, replicated data generated under the model should look similar to observed data. To put it another way, the observed data should look plausible under the posterior predictive distribution (Gelman et al. 2013).

- A self-consistency check.
- A data-informed data generating model.

Posterior predictive checks

The posterior predictive distribution:

$$p(y^{rep}|y) = \int p(y^{rep}|\theta)p(\theta|y)d\theta$$

- Fit a model $p(\theta|y)$ to an observed sample y.
- Draw S simulated values y^{rep} ("replications") from the joint posterior predictive distribution $p(y^{rep}|y)$.
- Define a statistic $T(y)$ that measures the discrepancy between model and data.
- Compare the generated samples to the observed data.

Any systematic differences between the simulations and the data indicate potential failings of the model.

How to chose the quantity T

A model can fail to reflect the data generating process in any number of ways. Compare a variety of statistics to evaluate more than one possible model failure (Gelman et al. 2013).

- Choose a quantity that reflects aspects relevant to the **scientific purpose**.
- Especially useful to measure features of data **not directly addressed by the model** (ex. ranks, correlations, relationships with explanatory variables).
- **Discard sufficient statistics** because we look for features not explicitly included in the model. Choose statistics that are orthogonal to model parameters instead.

How to compare the quantity T

- Numerically:
  - Compare the **magnitude** $T(y, \theta^s) - T(y^{\text{rep s}}, \theta^s)$.
  - Compute the **probability** of the replicated data being more extreme than the observed data $\Pr\left(T(y, \theta^s) \leq T(y^{reps}, \theta^s)\right) \ \forall \ s = 1, \ldots, S$.
- Graphically:
  - Histogram of $T(y, \theta^s) - T(y^{\text{rep s}}, \theta^s)$ should include zero.
  - Scatterplot $T(y, \theta^s) \sim T(y^{\text{rep s}}, \theta^s)$ should be symmetric about the 45º line.

Posterior predictive check - An example

```
fit <- stan(
    file    = "stan/garch11.stan",
    data    = list(
      T      = length(y),      # Length of the series
      H      = 1,              # Produce one-step ahead forecast
      y      = as.numeric(y), # The observed series (daily log-returns)
      sigma1 = sd(y)           # The conditional volatility at t = 1
    ),
    chains  = 4,
    iter    = 500,
    warmup  = 250,
    seed    = 9000,
    verbose = FALSE
  )

shat  <- apply(extract(fit, pars = "sigma")[[1]], 2, median)
spred <- extract(fit, pars = "spred")[[1]]
ypred <- extract(fit, pars = "ypred")[[1]]
sfore <- extract(fit, pars = "sfore")[[1]]
yfore <- extract(fit, pars = "yfore")[[1]]
```

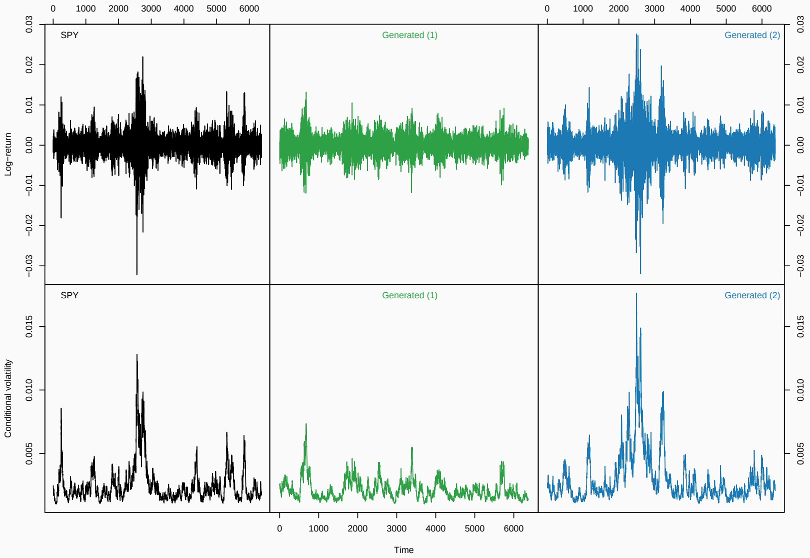
Posterior predictive check - An example

```
transformed parameters {
  real<lower=0> sigma[T];
  sigma[1] = sigma1;
  for (t in 2:T) {
    sigma[t] = sqrt(
      alpha0
      + alpha1 * pow(y[t-1], 2)
      + beta1 * pow(sigma[t-1], 2)
    );
  }
}

model {
  // Priors
  alpha0 ~ normal(0, 0.5) T[0, ]; // close to zero and small
  alpha1 ~ beta(0.50, 1.50);      // slightly more likely to be close to zero
  beta1  ~ beta(2.50, 0.80);      // slightly more likely to be close to one

  // Sampling (likelihood)
  y ~ normal(0, sigma);
}
```

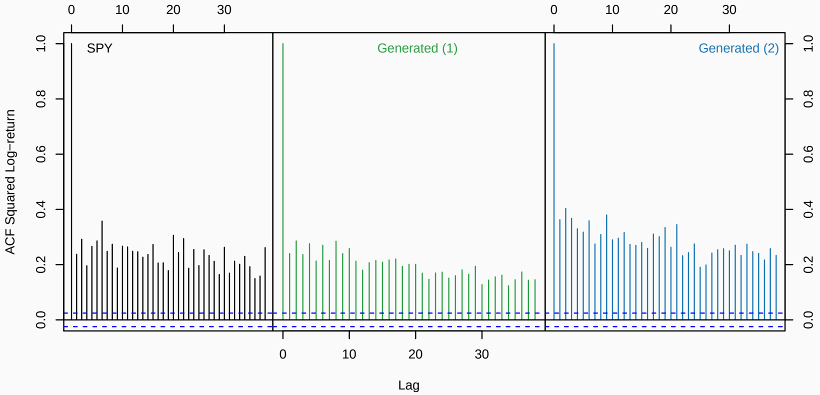
Posterior predictive check - An example

```
generated quantities {
  real<lower=0> sfore[H];
  real<lower=0> spred[T];
  real yfore[H];
  real ypred[T];

  for (h in 1:H) {
    sfore[h] = sqrt(
      alpha0
      + alpha1 * pow(y[T + h - 1], 2)
      + beta1 * pow(sigma[T + h - 1], 2)
    );

    yfore[h] = normal_rng(0, sfore[h]);
  }

  spred[1] = sigma1;
  ypred[1] = normal_rng(0, sigma1);
  for (t in 2:T) {
    spred[t] = sqrt(
      alpha0
      + alpha1 * pow(ypred[t-1], 2)
      + beta1 * pow(spred[t-1], 2)
    );
    ypred[t] = normal_rng(0, spred[t]);
  }
}
```

Observed vs. posterior predictive returns

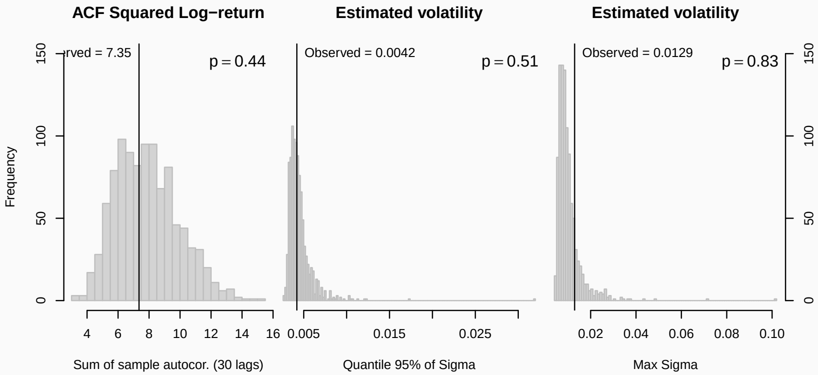


Observed vs. posterior predictive quantities (1)



Does our model replicate data features that are explicitly modeled?

Observed vs. posterior predictive quantities (2)



Does our model replicate data features that are **not** explicitly modeled?

Out-of-sample forecast evaluation

Methodology for **walking-forward validation**:

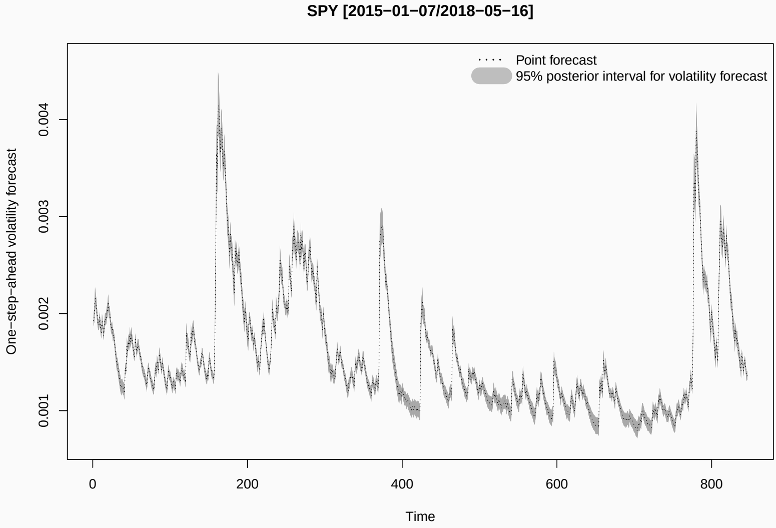
- **Extract subsamples** using a rolling window with 2520 observations each (approximately ten trading years).
- **Estimate the parameters** θ for each daily subsample.[2]
- Compute the h-step ahead **forecast for volatility** $\hat{\sigma}_{t+h|t}$.
- Draw a sample from the **expected distribution of log-returns** $\hat{y}_{t+h|t} \sim \mathcal{N}(0, \hat{\sigma}^2_{t+h|t})$.

Outcome:

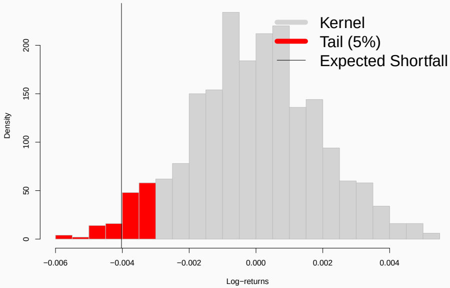
- Volatility forecasts (accounting for **parameter uncertainty**!)
- A full distribution of forecasted log-returns (accounting for parameter uncertainty as well).

[2] In some settings, you may consider fitting the model each k steps.

Parameter uncertainty



Distribution of expected return

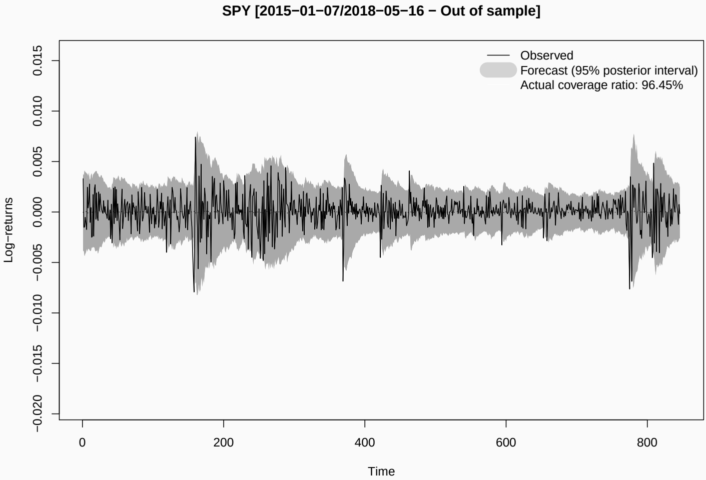


We draw an arbitrary large sample of expected log-returns $\hat{y}_{t+1|t}$ and use this empirical distribution to analyze risk:

- Mean/median log-returns.
- Standard deviation, interquartile range.
- Kurtosis.
- Quantiles (ex. VaR).
- Tail characteristics (ex. conditional mean for ES).

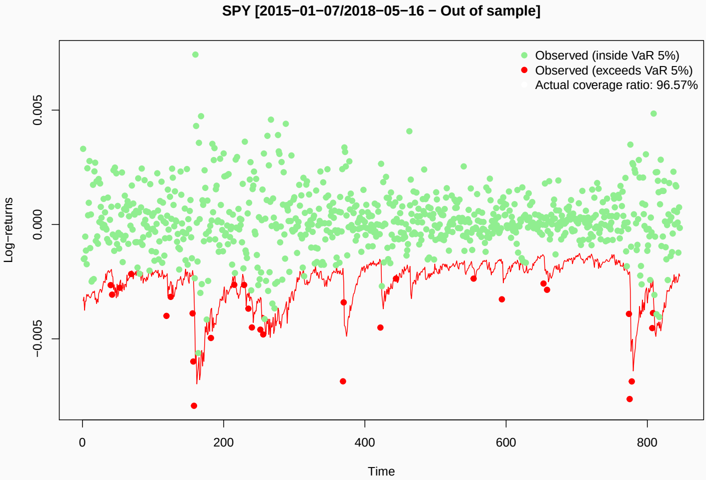
Coverage ratio

How many times does the (ex-post) observed log-return exceed our $1 - \alpha$ forecast interval?



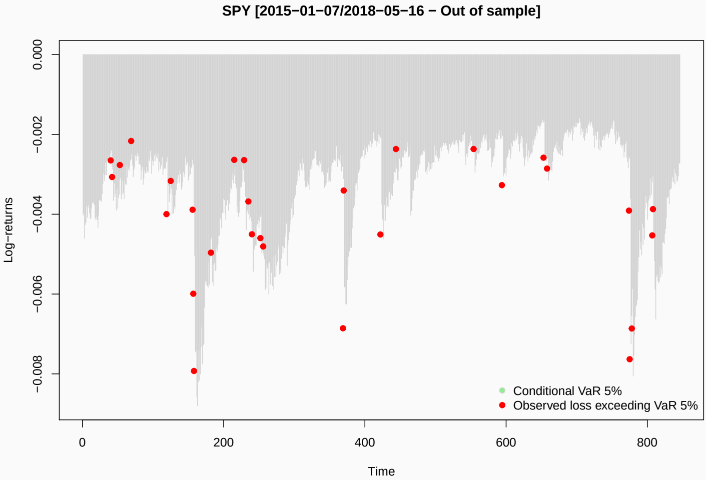
Value at Risk

How many times does the (ex-post) observed log-return exceed our α VaR?



Expected Shortfall

How does the (ex-post) observed log-return compare with our α ES?



Recap (1)

We studied the minimum suggested steps in the Bayesian workflow:

- State your scientific problem.
- Gather substantive knowledge (literature review, previous empirical studies, ask experts).
- Use exploratory data analysis to hypothesize about possible sources of heterogeneity.
- Set up a generative model (in Stan).
    - Parameter constraints and priors only, no observations yet.
    - Use priors to express beliefs and substantive knowledge, and to regularize and stabilize estimates.
    - Does data generated by the model produce the sought features?

Recap (2)

- Software validation
  - Now consider the likelihood: add sampling statements to your Stan code.
  - Generate "fake data" compliant with model specification, if possible using a different implementation (a "unit test" of sorts).
  - Does the model systematically recover the true parameters?
  - Are the estimates pausible and reasonable?
  - Do posterior intervals achieve the adequate coverage?
- Consider the sample to estimate the parameters and other hidden quantities.

Recap (3)

- Diagnose your model
  - MCMC: assess mixing with the $\hat{R}$ statistic, the effective sample size, traceplots and divergences.
  - Sensitivity analysis: how much posterior inference change when other reasonable priors and/or models are used?
  - Use posterior predictive checks: does the model replicate data features that are explicitly modeled? does it replicate features **not** modeled?
  - External validation (out-of-sample prediction): check coverage using walking-forward validation.

Next steps?

If you enjoyed the seminar, you should be reading. . .

- More about covered topics:
  - Stan (Team 2017; Carpenter et al. 2017).
  - Bayesian Inference (Gelman et al. 2013; McElreath 2015).
  - Volatility Models (Alexander 2008; Tsay 2010; Christoffersen 2016).
- Moving forward:
  - Bayesian Time Series, State Space Models, Bayesian Filtering (Prado and West 2010; Särkkä 2013).
  - Bayes & Risk (Jacquier, Polson, and Rossi 2002; Lopes and Tsay 2010).

References

Alexander, Carol. 2008. *Market Risk Analysis, Practical Financial Econometrics (Volume Ii)*. Wiley.

Bollerslev, Tim. 1986. "Generalized Autoregressive Conditional Heteroskedasticity."

Carpenter, Bob, Andrew Gelman, Matthew D. Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. 2017. "Stan: A Probabilistic Programming Language." *Journal of Statistical Software* 76 (1). Foundation for Open Access Statistic. doi:10.18637/jss.v076.i01.

Christoffersen, Peter. 2016. *Elements of Financial Risk Management, Second Edition*. Academic Press.

Cont, R. 2001. "Empirical Properties of Asset Returns: Stylized Facts and Statistical Issues." *Quantitative Finance* 1 (2). Informa UK Limited: 223–36. doi:10.1080/713665670.

Cook, Samantha R, Andrew Gelman, and Donald B Rubin. 2006. "Validation of Software for Bayesian Models Using Posterior Quantiles." *Journal of Computational and Graphical Statistics* 15 (3). Taylor & Francis: 675–92.

Engle, Robert F. 1982. "Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation." *Econometrica* 50 (4). JSTOR: 987. doi:10.2307/1912773.

Gabry, Jonah, Daniel Simpson, Aki Vehtari, Michael Betancourt, and Andrew Gelman. 2017. "Visualization in Bayesian Workflow," September. http://arxiv.org/abs/1709.01449v4.

Gelman, Andrew. 2004. "Exploratory Data Analysis for Complex Models." *Journal of Computational and Graphical Statistics* 13 (4). Informa UK Limited: 755–79. doi:10.1198/106186004x11435.

Gelman, Andrew, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. 2013. *Bayesian Data Analysis, Third Edition (Chapman & Hall/Crc Texts in Statistical Science)*. Chapman; Hall/CRC.

Jacquier, Eric, Nicholas G Polson, and Peter E Rossi. 2002. "Bayesian Analysis of Stochastic Volatility Models." *Journal of Business & Economic Statistics* 20 (1). Informa UK Limited: 69–87. doi:10.1198/073500102753410408.

Lopes, Hedibert F., and Ruey S. Tsay. 2010. "Particle Filters and Bayesian Inference in Financial Econometrics." *Journal of Forecasting* 30 (1). Wiley: 168–209. doi:10.1002/for.1195.

McElreath, Richard. 2015. *Statistical Rethinking: A Bayesian Course with Examples in R and Stan (Chapman & Hall/Crc Texts in Statistical Science)*. Chapman; Hall/CRC.

Prado, Raquel, and Mike West. 2010. *Time Series: Modeling, Computation, and Inference (Chapman & Hall/Crc Texts in Statistical Science)*. Chapman; Hall/CRC.

Särkkä, Simo. 2013. *Bayesian Filtering and Smoothing (Institute of Mathematical Statistics Textbooks)*. Cambridge University Press.

Team, Stan Development. 2017. *Stan Modeling Language: User's Guide and Reference Manual*. Version 2.17.0.

Tsay, Ruey S. 2010. *Analysis of Financial Time Series*. Wiley.