A long-short term event memory state-space model for multi-party elections

Marcus Groß

2020-12-30

Abstract

State-space models are a popular choice for modelling voting intentions and election results using polling data. The presented multivariate state-space model attempts to go beyond random-walk or Kalman-filter approaches (with comparable performance to simple weighted survey averages) by introducing a long-short term event memory effect to the problem. This effect serves as a reasonable explanation for the observation that the vote share partially tends to reverse to the party's long-term trend after larger short-term movements. Any event influencing the vote share of a party is presumed to have a convex shape, and be decomposable into a short-term effect due to e.g. media spreading, and a smaller, long-term effect remaining despite overlay effects of new events and forgetfulness. This effect is modelled by a mixture of a random walk and two contrasting autoregressive processes. By also taking advantage of the widely-observed effect that government parties tend to fall in vote share, whereas the opposite effect is observed for opposition parties, mid- and long-term predictions of election outcomes can be considerably improved. The Stan-model is fitted and evaluated on polling data from seven pollsters of the German national elections ("Bundestagswahl") from 1994 to 2017, where low double-digit (out-of-sample) improvements in prediction performance can be seen between 3- and 18-months prior to the elections. By taking into account the pollsters' house effects, their poll errors, and even more importantly, their correlations in poll errors, an appropriate and realistic estimation error can be created.

Data

The data amounts to more than 4,000 polls from seven different pollsters between November 1st, 1994, through the current date for the German federal election ("Bundestagswahl"). These data are scraped from the web page www.wahlrecht.de, which collects all available polling data and is frequently updated. Furthermore, election outcome data for all elections since 1998, and the respective parties forming the government and opposition is available. Data are available for the six large parties: CDU/CSU, SPD, Die Grünen, FDP, Die Linke, and AfD.

Get Poll and Election Data

First, a prediction date – March 25th, 2017 – is set, exactly six months prior to the election.

```
predDate <- as.Date("2017-03-25")
```

The polling data for the German election, the "Bundestagswahl", is scraped from the web page wahlrecht.de.

```
require('lsTermElectionForecast')
require('rstan')
require('dplyr')
require('tidyr')
require('zoo')
require('xlsx')
```

```
dataDE <- loadDataDE(predDate)
knitr::kable(head(dataDE$pollData))
```

	Institut	Datum	CDU/CSU	SPD	GRÜNE	FDP	LINKE	AfD	Befragte	Sonstige
49	Allensbach	2017-02-22	0.330	0.305	0.08	0.070	0.080	0.085	1542	0.05
50	Allensbach	2017-01-26	0.360	0.230	0.09	0.070	0.095	0.115	1441	0.04
51	Allensbach	2016-12-22	0.355	0.220	0.10	0.075	0.095	0.105	1459	0.05
52	Allensbach	2016-11-16	0.340	0.230	0.11	0.075	0.090	0.105	1436	0.05
53	Allensbach	2016-10-20	0.330	0.220	0.12	0.075	0.090	0.125	1458	0.04
54	Allensbach	2016-09-22	0.335	0.240	0.11	0.070	0.070	0.125	1407	0.05

```
Elections <- dataDE$Elections
knitr::kable(Elections)
```

CDU.CSU	SPD	GRÜNE	FDP	LINKE	AfD	Sonstige	Year	Datum	Institut
0.351	0.409	0.067	0.062	0.051	NA	5.9	1998	1998-09-27	Election
0.385	0.385	0.086	0.074	0.040	NA	3.0	2002	2002-09-22	Election
0.352	0.342	0.081	0.098	0.087	NA	4.0	2005	2005-09-18	Election
0.338	0.230	0.107	0.146	0.119	NA	6.0	2009	2009-09-27	Election
0.415	0.257	0.084	0.048	0.086	0.047	6.4	2013	2013-09-22	Election
0.329	0.205	0.089	0.107	0.092	0.126	5.0	2017	2017-09-24	Election
NA	NA	NA	NA	NA	NA	NA	2021	2021-09-24	Election

Motivation

The high levels of attention that Nate Silver's US Presidential election forecasts on fivethirtyeight.com have received demonstrate how data-based election forecasts are of great interest to the public, as well as in academia. There have been attempts to model elections using *STAN*, which rely on state-space models where the state (voter's intention) is modeled by a random walk. This approach follows the 2005 paper by Simon Jackman (https://www.tandfonline.com/doi/abs/10.1080/10361140500302472) and was used to predict the Australian election (cf. http://freerangestats.info/blog/2017/06/24/oz-polls-statespace), for example. While this method gives valuable insights and does a good job of removing bias from different pollsters, in the authors experience it is not superior to very simple poll-averaging methods for mid- or long-term forecasts. The question here is whether one can do better than just taking the current latent state or voter's intention as forecast for the actual election that may be six months or one year in the future.

Other election forecasts, such as those for the US Presidential election by Nate Silver (https://fivethirtyeight.com/features/a-users-guide-to-fivethirtyeights-2016-general-election-forecast/), multi-party forecasts for the UK election (http://www.electionforecast.co.uk), or the German federal election (http://zweitstimme.org) state that there is some form of mean-reversion in polls or election results in the long-term, and incorporate this into their election forecast in one way or another. For example, http://www.electionforecast.co.uk states that:

> The basic principle is that polling has some systematic biases, in particular a tendency to overstate changes from the previous election

FiveThirtyEight states that:

> Empirically, using more smoothing early in the race and less smoothing late in the race works best. In other words, the trend line starts out being quite conservative and becomes more aggressive as Election Day approaches.

There is some discussion in a 1993 paper by Andrew Gelman and Gary King on this issue (https://gking.harvard.edu/files/abs/variable-abs.shtml). In a 2013 paper by Drew Linzer (https://votamatic.org/wp-content/uploads/2013/07/Linzer-JASA13.pdf), a mean-reversion effect is incorporated for US Presidential elections by implicitely assuming that on the state level, the vote share is going to return to its long-term mean. However, such a form of mean stationarity is unrealistic, particularly in multi-party systems, as there are clear trends over time for different parties. For example, formerly successful parties can disappear entirely. Therefore, a mixture between a non-stationary random walk and a mean-reversing stationary process for the vote share over time is proposed.

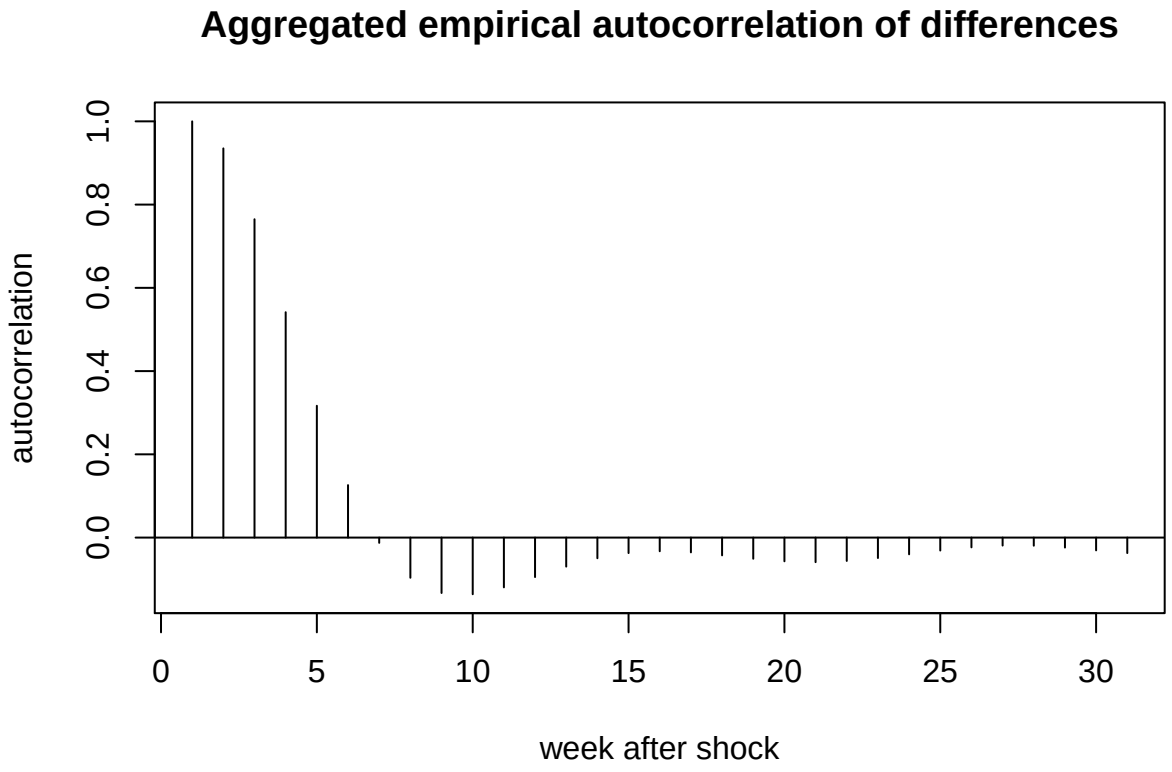
How can this kind mean-reversion process be interpreted? The idea is to model this as a long-short term memory process. Assuming weekly data and weekly changes, it is stated that each shock, or change in voter's intention for a party, is attributable to events, e.g. political scandals, controversial statements, or candidate selection. The initial shock, which can be positive or negative, is covered by the media in the following weeks, increasing the initial effect. After a few weeks the event or news might slowly disappear out of the public's mind, but won't entirely. The remaining effect accounts for the long-term effect of the initial event.

In contrast, FiveThirtyEight, for example, follows an approach where economic factors are used for forecasting and weighted with poll data forecasts. The proposed approach follows a different philosophy. It can be argued similar to the market efficiency hypothesis in the equity market all (economic,..) information is already incorporated in the current voters intention. Short term fluctuations of polls long before election are mitigated by the long-/short term memory model, while for models with economic factors this is done by weighting factor forecasts with polls. While both approaches yield similar results, for the presented one there is less danger of overfitting and variable selection bias.

Is there evidence for a long-/short memory of voters in the data on the German federal election? To investigate empirically, the polls are smoothed on a weekly basis using a smoothing spline, compute the autocorrelation functions (ACFs) on the weekly differences, and average the ACFs on all parties, excluding AfD (as this right-wing party appeared only a few years ago):

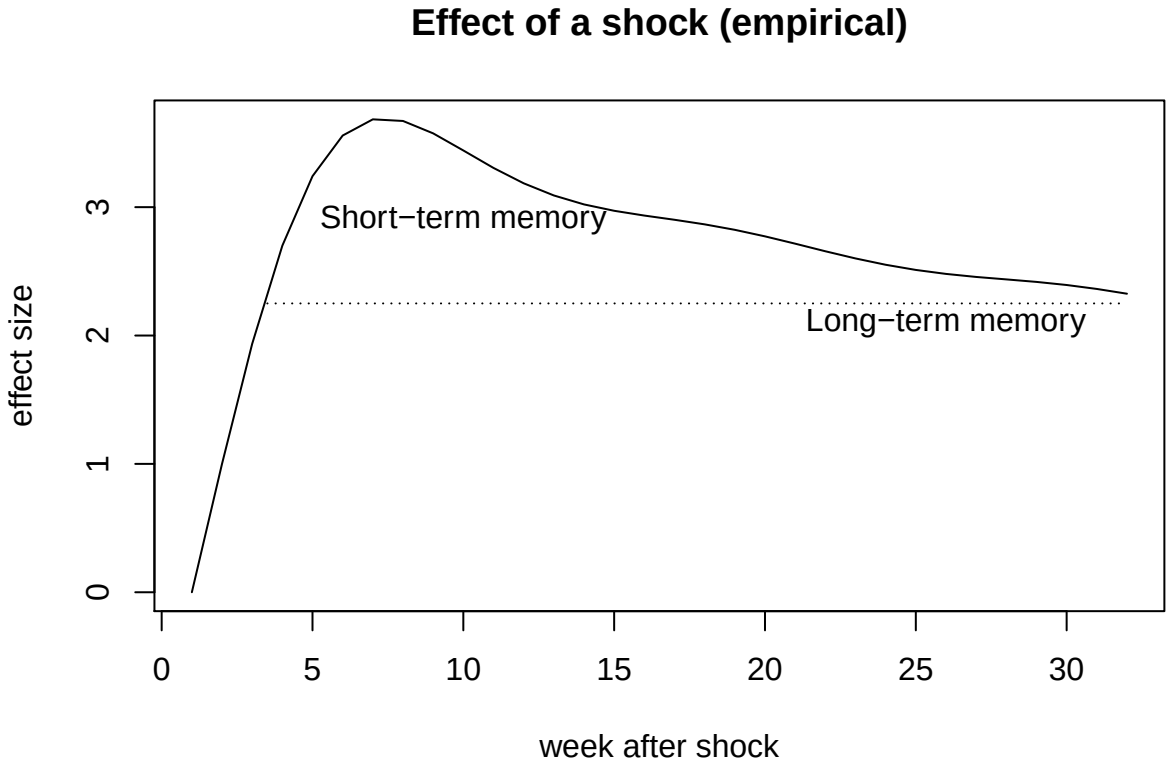
```
pollDataShock <- dataDE$pollData %>% arrange(desc(Datum)) %>% na.locf(na.rm = FALSE) %>%
  na.locf(fromLast = TRUE)
dateSeq <- seq(min(pollDataShock$Datum), max(pollDataShock$Datum) , by = "week")
smoothProportions <- sapply(colnames(pollDataShock)[3:7],
                            function(y){
                              sSpline <- smooth.spline(x = as.numeric(pollDataShock$Datum),
                                                       y = unlist(pollDataShock[,y]), cv = TRUE, nknots
                              predict(sSpline, x = as.numeric(dateSeq))$y
                            })

acfs <- rowMeans(apply(apply(smoothProportions, 2, diff), 2,
                       function(x) acf(x, 30, plot = FALSE)$acf))
plot(acfs, type = "h", xlab = "week after shock", ylab = "autocorrelation",
     main = "Aggregated empirical autocorrelation of differences")
abline(h = 0)
```



One can clearly see that there is a positive autocorrelation within the first weeks, turning negative after about 8 weeks after the initial shock. The first interval can be seen as a news / event spreading effect, where the initial impact is spilled over to the following weeks. The latter interval can be seen as the period where the events get out of the short-term public memory. A clearer picture of the mean-reversing nature of polls can be seen when the autocorrelation function is integrated:

```
plot(c(0,cumsum(acfs)), type = "l", xlab = "week after shock", ylab = "effect size",
     main = "Effect of a shock (empirical)")
segments(x0 = 3.5, x1 = 32, y0 = 2.25, y1 = 2.25, lty = 3)
text(x = 26, y = 2.1, "Long-term memory")
text(x = 10, y = 2.9, "Short-term memory")
```



Another peculiarity of elections over time is the effect of government parties generally losing vote share, while opposition parties tend to gain voters. For the German federal elections between 1994 and 2017, the government party lost vote share in 10 of 12 cases in the following election, and the opposition parties gained vote share in 15 of 20 cases. The same observation can be seen in neighboring countries with similar multi-party election systems such as the Netherlands or Sweden. This may be influenced by the fact that government parties are more likely to face negative events, whereas opposition parties experience more positive events.

Central to any election forecast is the realistic assessment of the forecast error. Two different sources of uncertainty in the forecast are considered:

1. Uncertainty about future events, i.e. shocks to vote share
2. Uncertainty in polling

The second point can be further split in three sources:

1. The common bias of all pollsters for a specific party and and time point
2. The (house) bias of a specific pollster for a specific party (and time point)
3. The polling uncertainty of a specific pollster for a specific party

The first source in particular is often completely ignored, and is also challenging to estimate. To illustrate, a look at the final polls just before the 2005 election and its outcome is taken:

```
knitr::kable(head(dataDE$pollData %>% arrange(desc(Datum)) %>% filter(Datum < as.Date("2005-09-18"))))
```

Institut	Datum	CDU/CSU	SPD	GRÜNE	FDP	LINKE	AfD	Befragte	Sonstige
Allensbach	2005-09-16	0.415	0.325	0.070	0.080	0.085	NA	1682	0.025
Allensbach	2005-09-13	0.417	0.329	0.072	0.070	0.085	NA	2000	0.027
Kantar(Emnid)	2005-09-13	0.420	0.335	0.070	0.065	0.080	NA	2000	0.030
Forsa	2005-09-12	0.420	0.350	0.070	0.060	0.070	NA	2504	0.030
GMS	2005-09-12	0.420	0.330	0.080	0.070	0.070	NA	1008	0.030

Institut	Datum	CDU/CSU	SPD	GRÜNE	FDP	LINKE	AfD	Befragte	Sonstige
Kantar(Emnid)	2005-09-10	0.405	0.345	0.070	0.070	0.080	NA	NA	0.030

```
knitr::kable(Elections %>% filter(Year == 2005))
```

CDU.CSU	SPD	GRÜNE	FDP	LINKE	AfD	Sonstige	Year	Datum	Institut
0.352	0.342	0.081	0.098	0.087	NA	4	2005	2005-09-18	Election

For the largest party, the CDU/CSU, all polls give an estimate that is much too high, exceeding the real outcome by between five and seven percentage points. Such large changes virtually never happen within one week, such that in this and many other cases, one can strongly presume that there is a common bias for all pollsters. This kind of bias will be adjusted for in the pollsters' post-processing, such that it will be approximately zero just after the election.

The house bias of a pollster for a certain party can be split in a time persistent and a fluctuating part.

For the third source, one may presume that the sample size ("Befragte" in the poll data) is an important factor, or even the only factor, for determining the uncertainty of a poll, but in the authors experience the effect is negligible. This is because the pollsters have their own correction methods such that the actual poll outcome and the published polls differ substantially.

Model

To model the vote share over time a "long-short term memory" state-space model is implemented in *STAN*. It incorporates a parameterized long- and short-term memory structure based on weekly events or shocks, incorporates the party status (government or opposition), accounts for different sources of uncertainty (future events, common pollster error, pollster-party-house bias, polling uncertainty), and heavy-tail errors.

Let $y_{party,t}$ be the true vote share for a specific party, pt for a certain time point t, if an election would be held, whereby t is indexed on a weekly basis. One cannot measure the vote share directly except on election days, so have to rely on polling data. The polling data $poll_{pt,t,pll}$ for a party pt, time point t and pollster pll is published in irregular intervals by different institutes or pollsters. The polling data and the election vote share were transformed on the logit scale to guarantee estimates in the (0, 1) interval after re-transformation, and to provide variance regulation:

$$logit(poll_{pt,t,pll}) \sim N(\mu_{pt,t,pll}\ , \sigma^2_{pt,pll} + 2E - 3)$$

The expectation of the transformed poll result corresponds to the vote share y and three forms of biases:

$$\mu_{pt,t,pll} = y_{pt,t} + bias_{pt,pll} + bias_{pt,pll,el} + \epsilon_{\text{pollError}_{pt,el}}$$

The pollsters' house bias and the pollsters' house bias by election come from a common distribution:

$$bias_{pt,pll} \sim N(0, \tau^2)$$

$$bias_{pt,pll,el} \sim N(0, \tau_2^2)$$

The pollsters common error $\epsilon_{\text{pollError}_{pt,el}}$ fluctuates over time. Its value is set to zero just after an election, as it is assumed that the pollsters adjust for the bias afterwards. As a weekly estimate is hard to identify, it was chosen to let it vary by election and party. It follows a multivariate normal distribution and accounts for covariance between parties:

$$\epsilon_{\text{pollError}_{pt,el}} \sim N(0, \sigma^2_{pt}\Sigma)$$

,with Σ as correlation matrix and σ_{pt} as party specific factor.

The standard deviation of a poll is a product of a pollster factor with the party standard deviation:

$$\sigma_{pt,pll} = \sigma_{pll}\sigma_{pt}$$

For an election, pollster specific bias and variance is non-existent. The upper expression gets simplified to:

$$logit(election_{pt,t}) \sim N(y_{pt,t}\ ,\ 2E-3)$$

The shifts in the common bias of the pollsters for all parties follow a multivariate t-distribution with 3.5 degrees of freedom and a standard deviation σ_{pollbias} that is different for each party pt multiplied by a correlation matrix Σ_{shift}:

$$\epsilon_{\text{polls}_{pt,t}} \sim mvt(0, \sigma^2_{\text{pollbias}_{pt}}\Sigma_{shift}, df = 3.5)$$

The true vote share $y_{pt,t}$ follows a random walk, but with three additional terms $\nu_{pt,t}$, $\eta_{1,pt,t}$ and $\eta_{2,pt,t}$, the short and long-term memory effects:

$$y_{pt,t} = y_{pt,t-1} + \epsilon_{pt,t} + \nu_{pt,t} - \eta_{1,pt,t} - \eta_{2,pt,t}$$

The shocks in vote share follow $\epsilon_{pt,t}$ follow a multivariate t-distribution with 3.5 degrees of freedom and a standard deviation $\sigma_{\text{shift}_{pt}}$ that is different for each party pt is multiplied by a correlation matrix. The expectation $\mu_{eps_{pt,t}}$ depends on the current state, i.e. state of being in government or opposition for the party.

$$\epsilon_{pt,t} \sim t(\mu_{eps_{pt,t}}, \sigma_{\text{shift}_{pt}}\Sigma_{shift}, df = 3.5)$$

As explained in the previous section, the expectation of the weekly shift in vote share might be influenced by whether a party is part of the government or the opposition.

$$\mu_{eps_{pt,t}} = opposition_{pt,t} + government_{pt,t}$$

The positive short-term memory parameter $\nu_{pt,t}$ follows a process resembling AR1. θ_2 marks the spreading speed of the weekly event effect. The higher, the lower the spreading speed of the news/event among the voters.

$$\nu_{pt,t} = \theta_2 \cdot (\nu_{pt,t-1} + \epsilon_{pt,t-1})$$

The diminishing short- and mid- term effect parameter $\eta_{1_{pt,t}}$ and $\eta_{2_{pt,t}}$ also follow a process resembling AR1. The additional parameters α and α_2 govern the amount of "forgetting" that takes place. For $\alpha = 1$, the long-term effect is zero, while for $\alpha = 0$, the long-term effect equals the short-term effect. The parameters θ_0 and θ_1 mark the short- and mid-term memory decay speed. They are fixed to 0.7 and 0.95, for half times of about 2 weeks and 3 months.

$$\eta_{1_{pt,t}} = \theta_0 \cdot \eta_{1_{pt,t-1}} + (1-\theta_0) \cdot \alpha \cdot \nu_{pt,t-1}$$

$$\eta_{2_{pt,t}} = \theta_1 \cdot \eta_{2_{pt,t-1}} + (1-\theta_1) \cdot \alpha_2 \cdot \nu_{pt,t-1}$$

Within *STAN*, the model, transformed parameters, and priors (mostly weakly informative, matched on the right scale) are the following. Many parameters were also generously interval-restricted, which turned out to be beneficial for fitting the model:

```
parameters {
  vector[NParties] y_start; // true vote share in first week
  real<lower=0, upper = 1> theta2; // weekly event spread AR1 parameter
  real<lower=0, upper = 1> alpha; //short term memory loss
  real<lower=0, upper = 0.8> alpha2; //short term memory loss
  //long term house bias party-pollster-combination
  vector<lower=-0.5, upper = 0.5>[NPollsters * NParties] housebias;
  //short term (election) house bias party-pollster-combination
  vector<lower=-0.5, upper = 0.5>[NElections * NPollsters * NParties] housebiasEl;
  real<lower=0, upper = 0.15> tau; //house bias sd
  real<lower=0, upper = 0.15> tau2; //house bias election sd
  real<lower=0, upper = 0.2> muPollster; // expectation of pollster sd
  real<lower=0, upper = 0.1> sdPollster; // sd of pollster sd
  // factor applied to pollster sd by party
  vector<lower=0, upper = 5>[NParties] sigma_sdParty;
  real<lower=0, upper = 0.15> mushift; // expectation of weekly shift sd
  real<lower=0, upper = 0.075> sdshift; // sd of weekly shift sd
  real<lower=0, upper = 0.3> mupbias; // expectation of common pollbias sd
  real<lower=0, upper = 0.175> sdpbias; // sd of common pollbias sd
  real<lower=-0.05, upper = 0.05> opposition; //opposition effect on weekly shift
  real<lower=-0.05, upper = 0.05> government; //government effect on weekly shift
  vector[NParties] epsilon[YTOTAL]; //weekly shift
  // common pollbias for party-election combination
  vector[NParties] pollError[NElections - 1];
  cholesky_factor_corr[NParties] Eps_corr; // correlation matrix of weekly shift
  cholesky_factor_corr[NParties] EpsPoll_corr; // correlation matrix of common poll bias
  vector[NPollsters] sigma_sdPollster_raw; // untransformed pollster sd
  vector[NParties] sigma_shift_raw; // untransformed weekly shift sd
  vector[NParties] sigma_pollbias_raw; // untransformed common pollbias sd
  vector<lower=0>[YTOTAL] u; // helper parameter for multivariate t
}
transformed parameters{
  vector[YTOTAL] y[NParties]; // true vote share: MAIN PARAMETER OF INTEREST
  vector[YTOTAL * NParties] w; // true vote share + common poll bias
  vector[YTOTAL] eps[NParties]; // weekly shift before memory effects
  vector[NTOTAL] mu; // poll expectation
  vector<lower=0>[NTOTAL] sigma; // poll sd
  vector[NPollsters * NParties] sigma_sd; // poll sd by party and pollster
  vector<lower=0>[NPollsters] sigma_sdPollster; // transformed common pollster sd
  vector<lower=0>[NParties] sigma_shift; // untransformed weekly shift sd
  vector<lower=0>[NParties] sigma_pollbias; // transformed common pollbias sd
  real eta; // short term memory loss current week
  real eta2; // mid term memory loss current week
  real nu; // spread effect current week
  sigma_shift = sigma_shift_raw * sdshift + mushift;
  sigma_pollbias = sigma_pollbias_raw * sdpbias + mupbias;
  sigma_sdPollster = sigma_sdPollster_raw * sdPollster + muPollster;

  for(i in 1:NParties){
    y[i,1]         =  y_start[i];
    eta = 0;
    eta2 = 0;
    nu = 0;
```

```
    eps[i] = ((to_vector(epsilon[,i]) .* sqrt(u)) +
                opposition + govMatrix[i] * government) * sqrt(1-square(theta2));
    sigma_sd[((i-1) * NPollsters + 1) : (i * NPollsters)] =
      (sigma_sdParty[i] / mean(sigma_sdParty)) * sigma_sdPollster;

    for(n in 2:YTOTAL){
      y[i,n]          = y[i,n-1]  + eps[i,n] + nu - eta - eta2;
      // short term memory loss (after fraction line: regularization)
      eta = eta * theta0 + alpha * nu * (1 - theta0) /
        sqrt(1-square(theta0)) * theta2;
      // mid term memory loss (after fraction line: regularization)
      eta2 = eta2 * theta1 + alpha2 * nu * (1 - theta1) /
        sqrt(1-square(theta1)) * theta2;
      nu = (eps[i,n] + nu) * theta2;
    }
    w[((i-1) * YTOTAL + 1) : (i * YTOTAL)] = y[i] +
      ElectionMatrix * to_vector(pollError[,i]);
  }
    mu = w[matchedDates] +
    csr_matrix_times_vector(NTOTAL, NPollsters * NParties,
                            Aw1, Av1, Au1, housebias) +
    csr_matrix_times_vector(NTOTAL, NPollsters * NParties * NElections,
                            Aw2, Av2, Au2, housebiasEl);
    sigma = csr_matrix_times_vector(NTOTAL, NPollsters * NParties,
                                    Aw1, Av1, Au1, sigma_sd) + 0.002;
}
model {
  sigma_shift_raw ~ std_normal();
  sigma_pollbias_raw ~ std_normal();
  sigma_sdPollster_raw ~ std_normal();
  y_start ~ normal(-1, 3);
  government ~ normal(0, 0.005);
  opposition ~ normal(0, 0.005);
  alpha ~ normal(0, 0.5);
  alpha2 ~ normal(0, 0.25);
  theta2 ~ normal(0.5, 0.25);
  tau ~ normal(0, 0.05);
  tau2 ~ normal(0, 0.05);
  mushift ~ normal(0.02, 0.01);
  mupbias ~ normal(0.1, 0.075);
  sdshift ~ normal(0, 0.01);
  sdpbias ~ normal(0, 0.03);
  Eps_corr ~ lkj_corr_cholesky(3);
  EpsPoll_corr ~ lkj_corr_cholesky(3);
  muPollster ~ normal(0.1, 0.05);
  sdPollster ~ normal(0.05, 0.025);
  u ~ scaled_inv_chi_square(nuT, 1);
  epsilon ~ multi_normal_cholesky(Zero,
                                  diag_pre_multiply(sigma_shift, Eps_corr));
  pollError ~ multi_normal_cholesky(Zero2,
                                    diag_pre_multiply(sigma_pollbias, EpsPoll_corr));
  sigma_sdParty ~ normal(1, 0.5);
  housebias ~ normal(0, tau);
```

```
  housebiasEl ~ normal(0, tau2);
  pollData ~ normal(mu, sigma);
}
```

Data Preparation

Before the model is compiled and sampled from, the polling and election data are bound together into a single data frame. A dummy matrix of the pollsters and parties ("IMatrix"), pollsters parties and elections ("IMatrixEl"), indicators on whether a certain observation of the combined data is a poll a certain election ("ElectionMatrix") is formed as well. Additionally, the weekly time sequence of the underlying state (vote share) with the time of the actual polls ("matchedDates") is matched and a matrix that assigns the government/opposition state for each party and point in time ("govMatrix") is build.

```
dataPrep <- preparePollData(dataDE$pollData, dataDE$Elections, predDate)
```

After preparation the result is transferred to stan, where for performance reasons a sparse representation of dummy matrices is computed:

```
data {
  int<lower=0> NTOTAL; // number of poll results (one for each poll and party)
  int<lower=0> NElections; // number of elections (including upcoming)
  int<lower=0> YTOTAL; // number of weeks modeled
  int<lower=0> NPollsters; // number of pollsters
  int<lower=0> NParties; //  number of parties
  int matchedDates[NTOTAL]; // matches polls with week number
  vector[NTOTAL] pollData; // poll results
  // dummy matrix party-pollster interaction for poll assignment
  matrix[NTOTAL, NPollsters * NParties] IMatrix;
  // dummy matrix party-pollster-election interaction for poll assignment
  matrix[NTOTAL, NElections * NPollsters * NParties] IMatrixEl;
  // dummy matrix for each party and week combination gov = 1,
  // opposition = 0; for week assignment
  vector[YTOTAL] govMatrix[NParties];
  // dummy matrix election for week assignment
  matrix[YTOTAL, NElections - 1] ElectionMatrix;
  //dummy variable upcoming election (0/1) for week assignment
  vector[YTOTAL] ElectionVector;
  //matrix of zeros for vote share shift by week expectation
  vector[NParties] Zero[YTOTAL];
  //matrix of zeros for poll error by week expectation
  vector[NParties] Zero2[NElections - 1];
}
transformed data {
  real theta0 = 0.7; // short term decay
  real theta1 = 0.95; // mid term memory decay
  real nuT = 3.5; // degrees of freedom for vote share shift parameter
  vector[rows(csr_extract_w(IMatrix))] Aw1;
  int Av1[rows(Aw1)];
  int Au1[size(csr_extract_u(IMatrix))];
  vector[rows(csr_extract_w(IMatrixEl))] Aw2;
  int Av2[rows(Aw2)];
  int Au2[size(csr_extract_u(IMatrixEl))];
```

```
  Aw1 = csr_extract_w(IMatrix);
  Av1 = csr_extract_v(IMatrix);
  Au1 = csr_extract_u(IMatrix);
  Aw2 = csr_extract_w(IMatrixEl);
  Av2 = csr_extract_v(IMatrixEl);
  Au2 = csr_extract_u(IMatrixEl);
}
```

Sampling with RStan

Now the model can be compiled and sample from. The quite complex nature of the model requires a tree depth of 13 or more. Together with the large number of parameters (several tens of thousands) it takes at least 12 hours to complete it, depending on the machine. For this report, several models were pre-computed at different points in time and the samples were saved previously.

```
#modelResults <- compileRunModel(dataPrep$modelData)
load("model_results/modelResults_250317.Rdata")
```

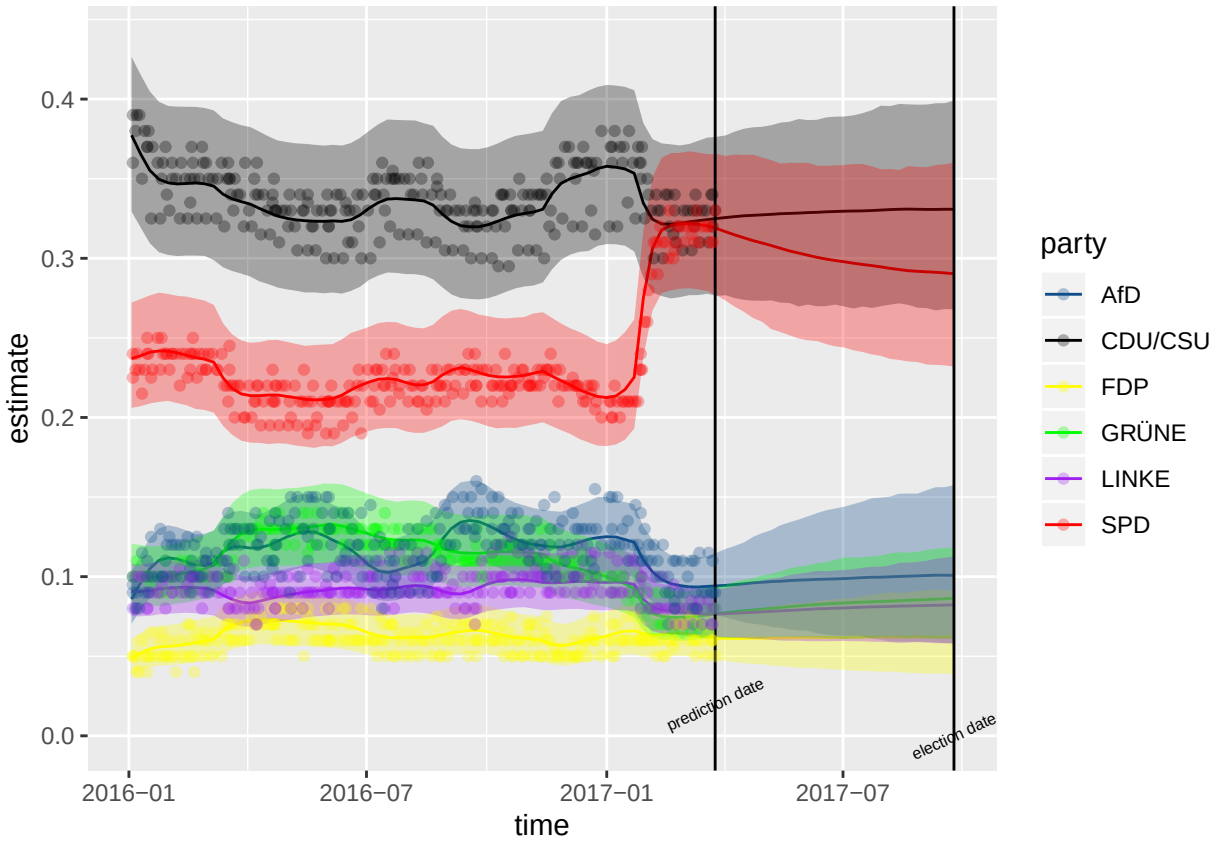
Results

Interpretation and visualization of model results

Prediction six months prior to the election

Six months before the election, the Social Democratic Party (SPD) experienced a large increase (from slightly over 20% to more than 30% in the polls) in vote share due to hype from the recently-announced Chancellor candidate Martin Schulz. From the picture below, one can clearly see a mean-reversing trend until the election date for the SPD (in red). The model estimates of the vote share for each party are presented as solid lines, with additional 95% credible bands from January 1st, 2016, until the date of election on September 24th, 2017. The date of prediction (vertical line) and the poll results (points) are also added:

```
plotForecast <- plotElectionData(modelResults, dataPrep, predDate,
                                 dataDE$pollData, start = "2016-01-01")
plotForecast$plot
```



To assess the predictive performance, the prediction is evaluated by the RMSE ("Root Mean Square Error"). As competitors or comparison the most recent poll and the average of the most recent poll of each pollster is used:

```
predElection <- getForecastTable(modelResults, dataPrep, predDate) %>%
  filter(type == "election_day")

#Model predictions:
predElection$party <- dataPrep$parties
knitr::kable(predElection[, c(1, 8, 4, 5, 6)])
```

date_forecast	party	estimate	lower_bound	upper_bound
2017-03-25	CDU/CSU	0.331	0.253	0.415
2017-03-25	SPD	0.290	0.219	0.377
2017-03-25	GRÜNE	0.087	0.057	0.127
2017-03-25	FDP	0.062	0.035	0.098
2017-03-25	LINKE	0.082	0.054	0.119
2017-03-25	AfD	0.101	0.052	0.175

```
#Recent polls:
knitr::kable(dataDE$pollData %>% filter(Datum <= predDate) %>%
  arrange(desc(Datum)) %>%
  group_by(Institut) %>% slice(1))
```

Institut	Datum	CDU/CSU	SPD	GRÜNE	FDP	LINKE	AfD	Befragte	Sonstige

Institut	Datum	CDU/CSU	SPD	GRÜNE	FDP	LINKE	AfD	Befragte	Sonstige
Allensbach	2017-02-22	0.33	0.305	0.080	0.070	0.080	0.085	1542	0.05
Forsa	2017-03-22	0.34	0.310	0.070	0.060	0.070	0.090	2504	0.06
Forsch'gr.Wahlen	2017-03-10	0.34	0.320	0.070	0.050	0.080	0.090	1212	0.05
GMS	2017-03-23	0.34	0.310	0.080	0.060	0.080	0.090	1008	0.04
Infratestdimap	2017-03-23	0.32	0.320	0.080	0.060	0.070	0.110	1023	0.04
INSA	2017-03-20	0.31	0.320	0.065	0.065	0.085	0.115	1933	0.04
Kantar(Emnid)	2017-03-25	0.33	0.330	0.080	0.050	0.080	0.090	2450	0.04

```
#Compute competitors
predRecent <- dataDE$pollData %>% filter(Datum <= predDate) %>%
  arrange(desc(Datum)) %>% select(dataPrep$parties) %>%
  slice(1) %>% unlist
predRecent
```

```
## CDU/CSU     SPD   GRÜNE     FDP   LINKE     AfD
##    0.33    0.33    0.08    0.05    0.08    0.09
```

```
predAvgRecent <- dataDE$pollData %>% filter(Datum <= predDate) %>%
  arrange(desc(Datum)) %>%
  group_by(Institut) %>% slice(1) %>% ungroup %>%
  select(dataPrep$parties) %>% colMeans %>% unlist %>% round(3)
predAvgRecent
```

```
## CDU/CSU     SPD   GRÜNE     FDP   LINKE     AfD
##   0.330   0.316   0.075   0.059   0.078   0.096
```

```
#RMSE model and competitors
sqrt(mean((predElection$estimate - unlist((dataDE$Elections %>%
                                             filter(Year == 2017))[, 1:6]))^2)) %>% round(4)
```

```
## [1] 0.0408
```

```
sqrt(mean((predRecent - unlist((dataDE$Elections %>%
                                  filter(Year == 2017))[, 1:6]))^2)) %>% round(4)
```

```
## [1] 0.0583
```

```
sqrt(mean((predAvgRecent - unlist((dataDE$Elections %>%
                                     filter(Year == 2017))[, 1:6]))^2)) %>% round(4)
```

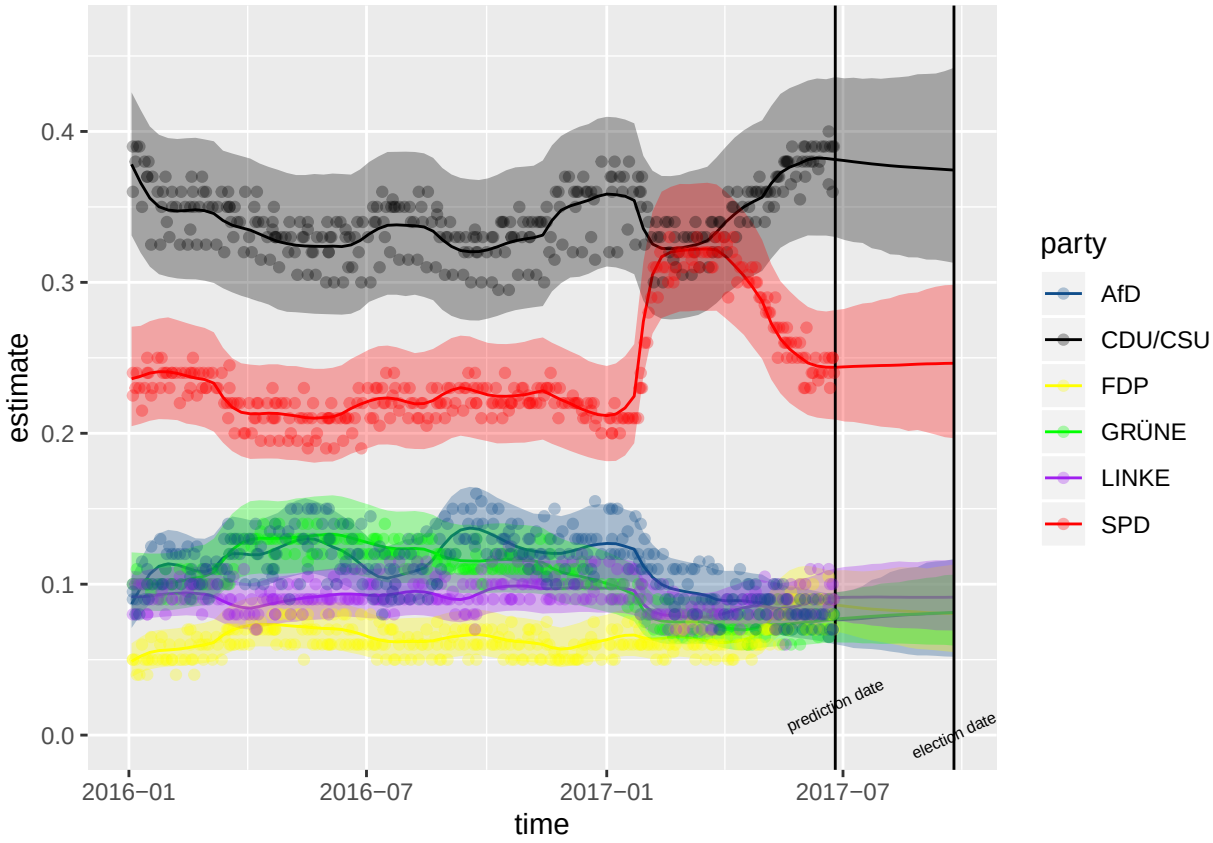
```
## [1] 0.0515
```

One can see that – at least at this point in time, six months before the election – the proposed model estimates beat the pollsters' by a fairly large amount, a 26% lower RMSE than the average of the recent predictions.

Now a look at a second prediction date, June 25th, 2017 – three months before the election is taken:

```
predDate <- as.Date("2017-06-25")
dataDE <- loadDataDE(predDate)
dataPrep <- preparePollData(dataDE$pollData, dataDE$Elections, predDate)
# modelResults <- compileRunModel(dataPrep$modelData)
load("model_results/modelResults_250617.Rdata")
```

```
plotForecast <- plotElectionData(modelResults, dataPrep, predDate,
                                 dataDE$pollData, start = "2016-01-01")
plotForecast$plot
```



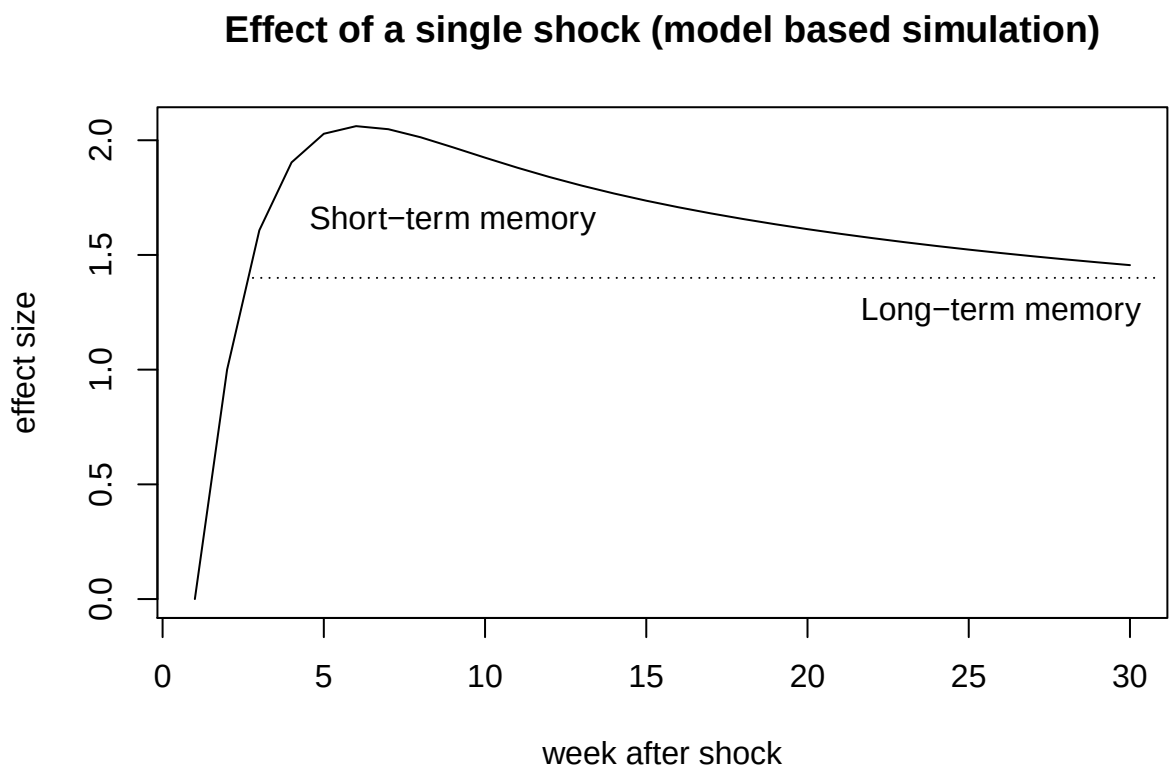
In the Motivation section the effect of a single shock in time with the model and the estimated parameters was simulated. One instantly sees that the graph is very similar to the empirical evidence on the model-free smoothed data in the Motivation section. A maximum effect at around seven to eight weeks after the initial shock, which partially wears off in the following weeks can be identified.

```
### effect of single shock of size 1
nu <- 0
eta <- 0
eta2 <- 0
weeks <- 30
y <- rep(0,weeks)
eps <- c(0, 1, rep(0,weeks-2))
theta0 =  0.7
theta1 =  0.95
theta2 = mean(modelResults$samples$theta2)
alpha = mean(modelResults$samples$alpha)
alpha2 = mean(modelResults$samples$alpha2)

for(n in 2:weeks){
  y[n]         = y[n-1]  + eps[n] + nu - eta - eta2;
  eta = eta * theta0 + alpha * nu * (1 - theta0) / sqrt(1-(theta0)^2) * theta2
  eta2 = eta2 * theta1 + alpha2 * nu * (1 - theta1) / sqrt(1-(theta1)^2) * theta2
  nu = (eps[n] + nu) * theta2
```

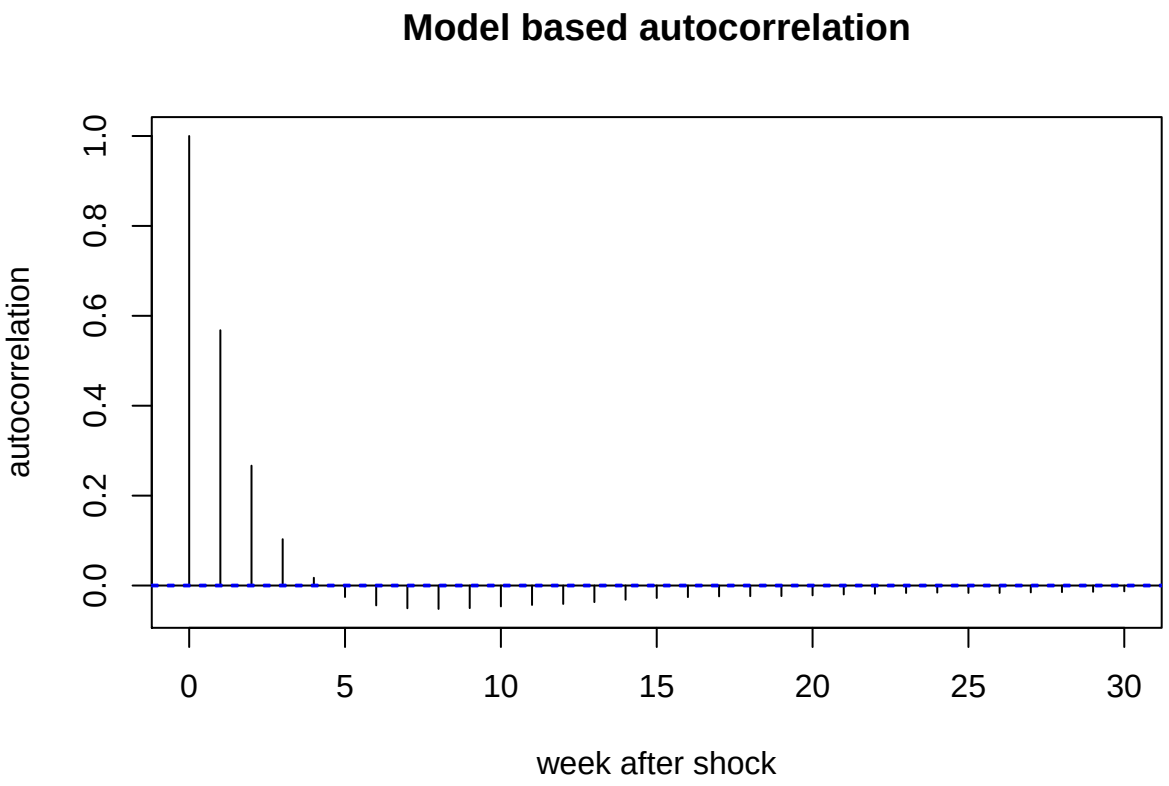
```
}

plot(y, type = "l", main = "Effect of a single shock (model based simulation)",
     xlab = "week after shock", ylab = "effect size")
segments(x0 = 2.8, x1 = 31, y0 = 1.4, y1 = 1.4, lty = 3)
text(x = 26, y = 1.25, "Long-term memory")
text(x = 9, y = 1.65, "Short-term memory")
```



One can also do a full simulation and compute the autocorrelation function. Again, the result is very similar to the results on the smoothed polling data in the Motivation section:

```
#Complete simulation; compute ACF
eps <- rt(1000000, df = 3.5)
y <- rep(0,1000000)
nu <- 0
eta <- 0
eta2 <- 0
for(n in 2:1000000){
  y[n]          = y[n-1]  + eps[n] + nu - eta - eta2;
  eta = eta * theta0 + alpha * nu * (1 - theta0) / sqrt(1-(theta0)^2) * theta2
  eta2 = eta2 * theta1 + alpha2 * nu * (1 - theta1) / sqrt(1-(theta1)^2) * theta2
  nu = (eps[n] + nu) * theta2
}
acf(diff(y), 30, xlab = "week after shock", ylab = "autocorrelation",
    main = "Model based autocorrelation")
```

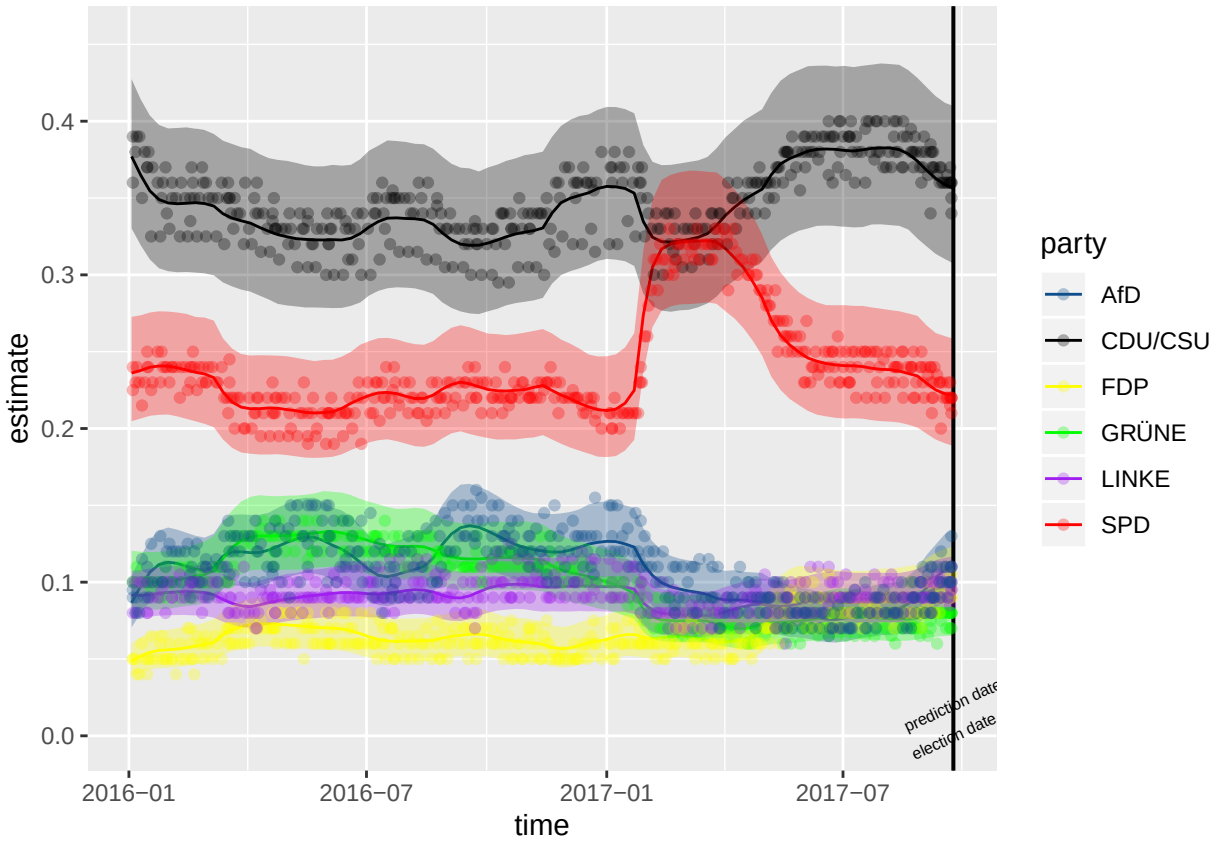


In the meantime, the uplift in vote share for the SPD has vanished almost completely, and the largest party, CDU/CSU (the party of chancellor Angela Merkel) gained some share. The model predicts a mean-reversing trend for the CDU/CSU until election (i.e. a decrease in vote share), but also for the AfD, where the model predicts an increase in vote share. It also becomes obvious that the AfD has much larger credible intervals compared to the other three smaller parties on election day. This might be due to the fact that this party was only founded a few years ago and is less stable, therefore experiencing larger swings in vote share.

Prediction the day prior to the election

Lastly, a closer look at the day before the election is taken:

```
predDate <- as.Date("2017-09-23")
dataDE <- loadDataDE(predDate)
dataPrep <- preparePollData(dataDE$pollData, dataDE$Elections, predDate)
# modelResults <- compileRunModel(dataPrep$modelData)
load("model_results/modelResults_230917.Rdata")
plotForecast <- plotElectionData(modelResults, dataPrep, predDate,
                                 dataDE$pollData, start = "2016-01-01")
plotForecast$plot
```

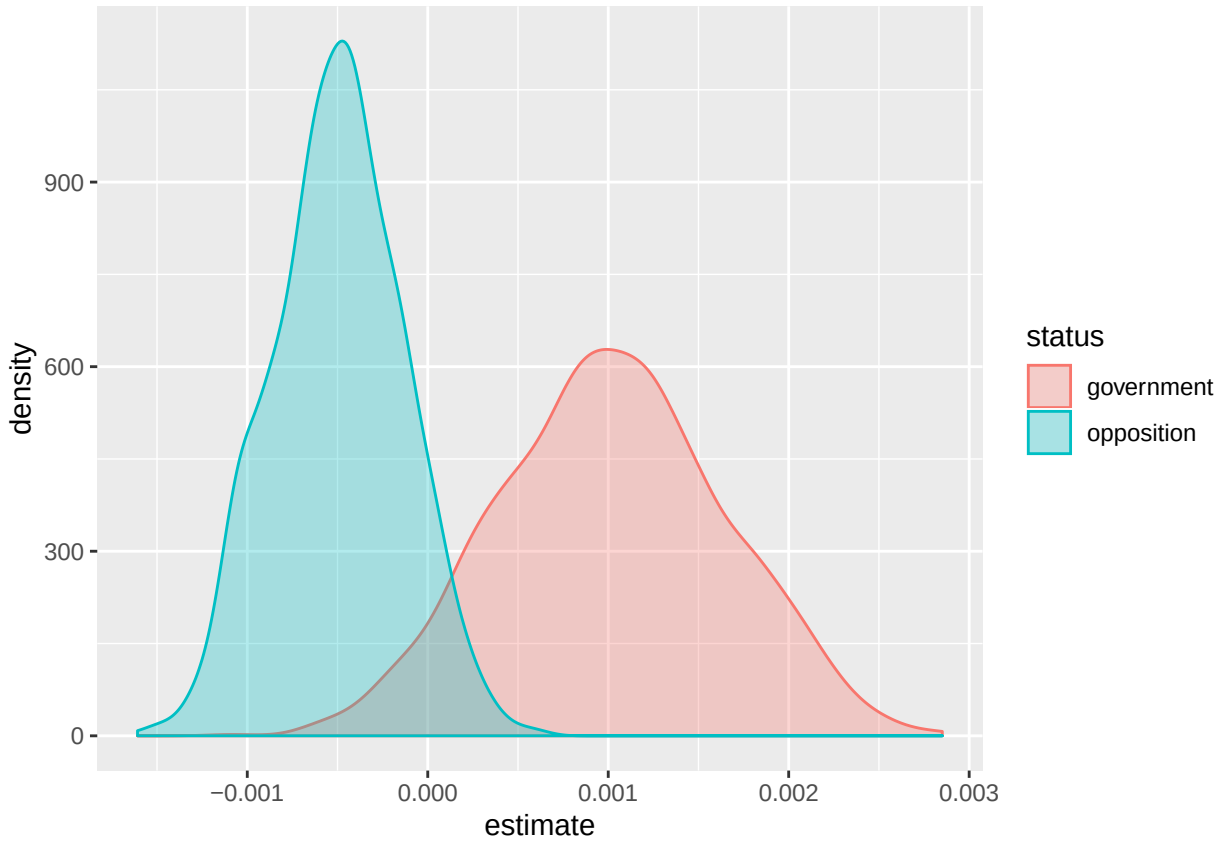


The CDU/CSU lost some vote share, while the AfD gained quite a bit, confirming the estimated trend from the model above.

Now, a look at the effect of government and opposition status on a party is taken. As presumed, being in opposition has a positive influence (in terms of expected value) on the shock, while being in government has a negative effect:

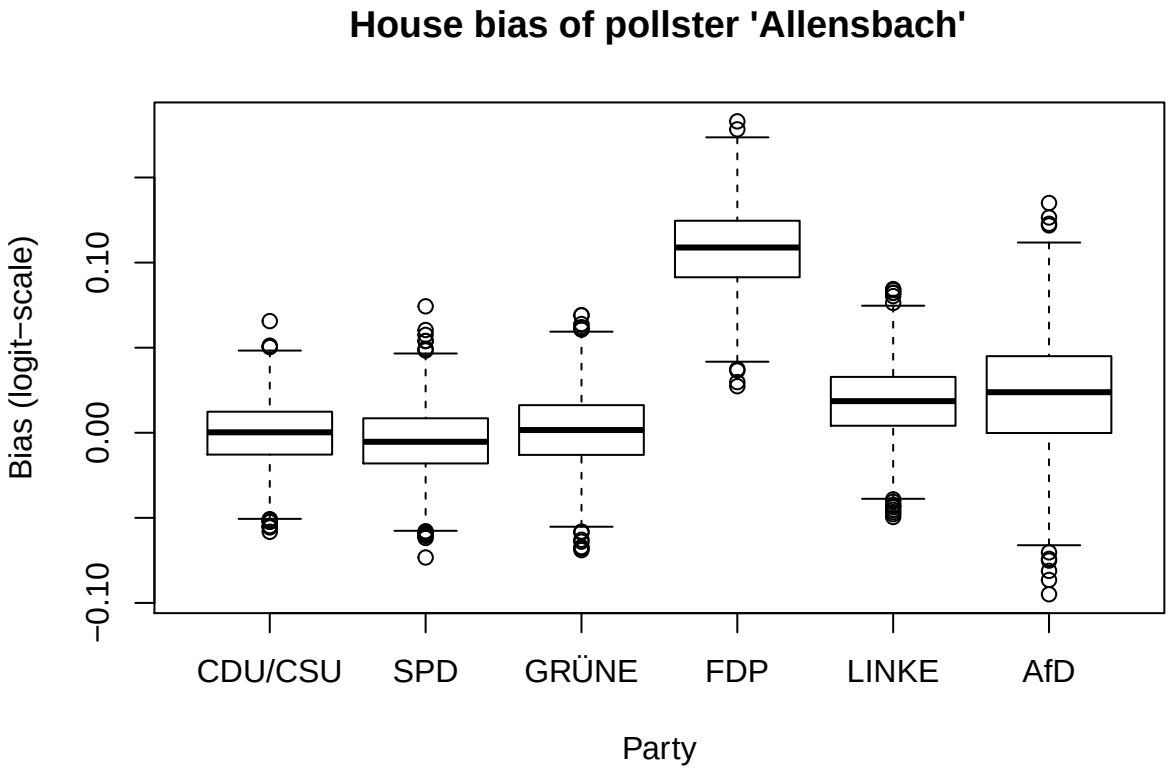
```
goData <- data.frame(estimate = c(modelResults$samples$opposition,
                                  modelResults$samples$government),
           status = rep(c("opposition", "government"),
                        each = length(modelResults$samples$government)))

ggplot(goData, aes(estimate, colour= status, fill = status)) + geom_density(alpha = 0.3)
```



The model allows us to also look at the "house bias" , i.e. the bias of a pollster on a specific party or the polling uncertainty. In the example of the "Allensbach" pollster, one sees that this pollster strongly favors the FDP:

```
biasAllensbach <- modelResults$samples$housebias[, grep("Allensbach",
                                                        colnames(dataPrep$modelData$IMatrix))]
colnames(biasAllensbach) <- dataPrep$parties
boxplot(biasAllensbach, xlab = "Party", ylab = "Bias (logit-scale)",
        main = "House bias of pollster 'Allensbach'")
```



Assessing predictive performance

To assess the performance of the proposed model, models on a monthly basis from 12 months before the election until the election in 2017 were computed. Now, the (out-of-sample) performance together with four competitors is evaluated:

1. The average on party-level of the most recent poll of all seven pollsters ("Average")
2. The most recent poll ("Recent")
3. The best pollster for the 2017 election, which is the pollster "INSA" ("Best")
4. The best pollster for each time point ("Min")

Note that competitors 3 and 4 have an "unfair" hindsight advantage, as they are not available until the election results are known.

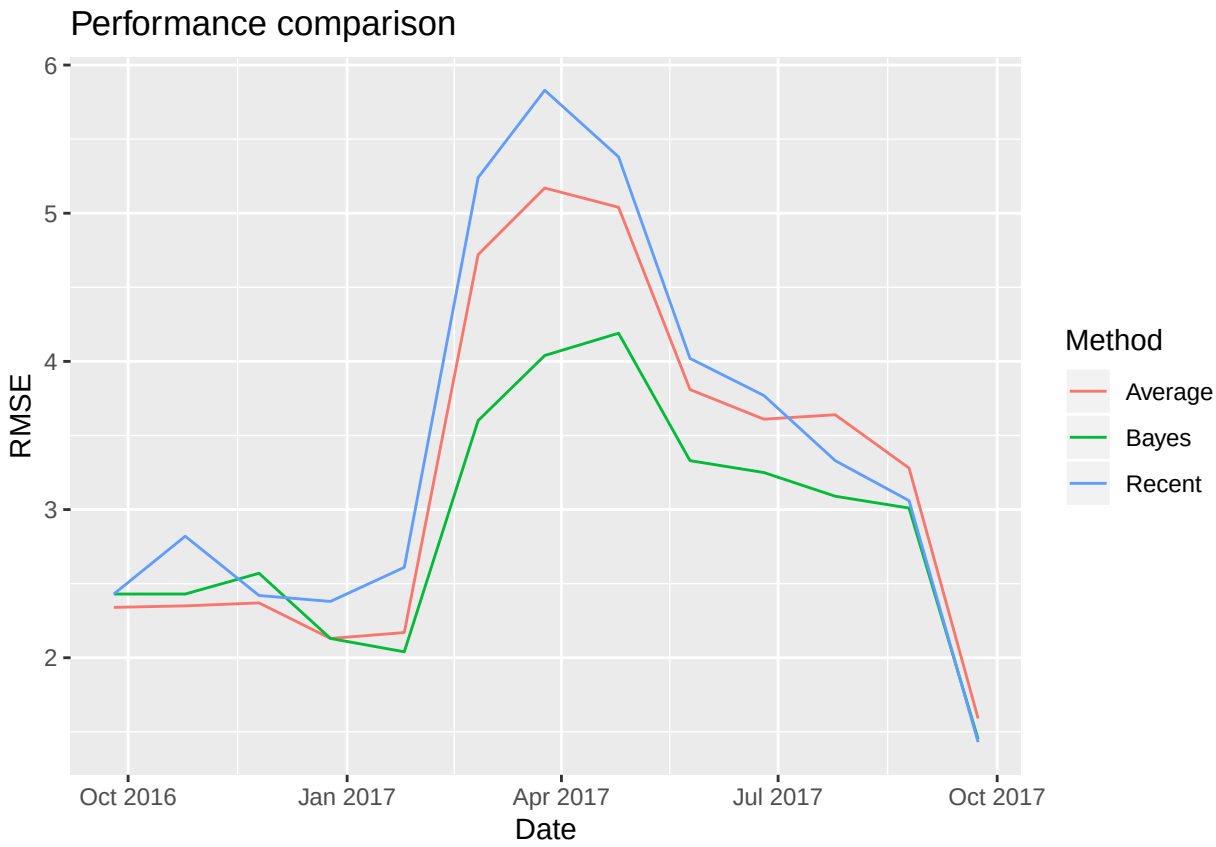
The Root Mean Square Errors averaged over all parties is stored in an excel file:

```
predictivePerformance <- read.xlsx("model_results/PredictivePerformance.xlsx",
                                   1, header=TRUE)
knitr::kable(predictivePerformance)
```

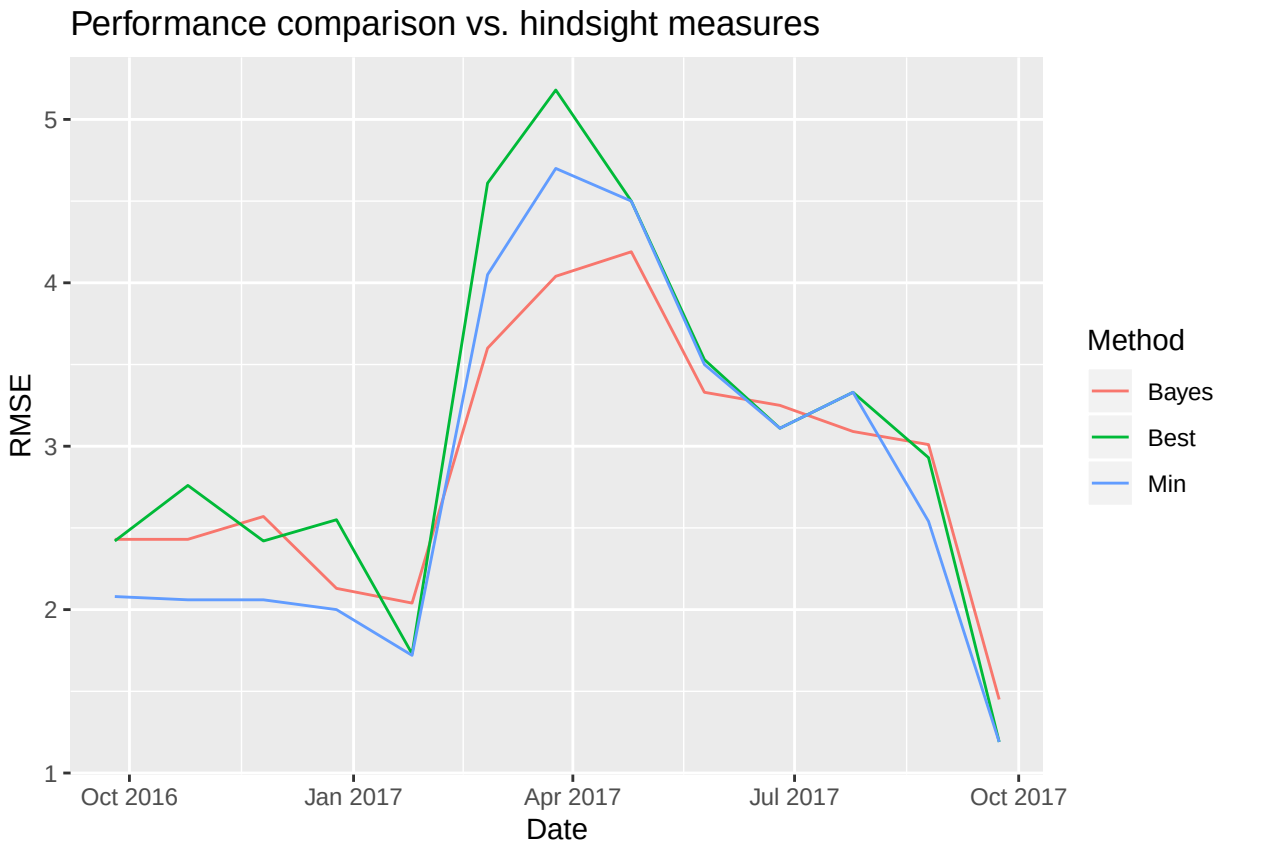
Date	Bayes	Average	Min	Best	Recent
2016-09-25	2.43	2.34	2.08	2.42	2.43
2016-10-25	2.43	2.35	2.06	2.76	2.82
2016-11-25	2.57	2.37	2.06	2.42	2.42
2016-12-25	2.13	2.13	2.00	2.55	2.38
2017-01-25	2.04	2.17	1.72	1.73	2.61
2017-02-25	3.60	4.72	4.05	4.61	5.24
2017-03-25	4.04	5.17	4.70	5.18	5.83
2017-04-25	4.19	5.04	4.50	4.50	5.38

Date	Bayes	Average	Min	Best	Recent
2017-05-25	3.33	3.81	3.50	3.53	4.02
2017-06-25	3.25	3.61	3.11	3.11	3.77
2017-07-25	3.09	3.64	3.33	3.33	3.33
2017-08-25	3.01	3.28	2.54	2.93	3.06
2017-09-23	1.45	1.59	1.19	1.19	1.43

```
longData <- gather(predictivePerformance, "Method", "RMSE", -Date)
ggplot(longData %>% filter(Method %in% c("Bayes", "Average", "Recent")),
       aes(y=RMSE, x= Date, colour = Method)) + geom_line() +
  ggtitle("Performance comparison")
```




```
ggplot(longData %>% filter(Method %in% c("Bayes", "Min", "Best")),
       aes(y=RMSE, x= Date, colour = Method)) + geom_line() +
  ggtitle("Performance comparison vs. hindsight measures")
```




```
#average over all time points
round(colMeans(predictivePerformance[,-1]), 3)
```

```
##   Bayes Average     Min    Best  Recent
##   2.889   3.248   2.834   3.097   3.440
```

The proposed Bayesian long-short term memory state-space model was able to beat a poll average by 15% on average over the year prior to election, and by almost 20% compared to the most recent poll. It even turned out to beat the best pollster for this particular election – which is only available in hindsight – and is almost on par with best pollster for each point in time, a remarkable result. Similar results are obtained when applying the model on the previous 2013 election.

Troubleshooting

The model iterations exceed maximum tree size, for tree_depth < 13. As each additional tree_depth increases the computation time about two-fold, this leads to slow computation. There were, very rarely, divergent samples. Therefore, the user might increase adapt_stepsize above the value of 0.8 in edge cases (the results are virtually identical, though). To allow for daily updates models has been running with rather low sample size (600 with 350 burn-in) in practice. For more precise results, multiple chains or longer sampling periods are desirable.

Government and Coalitions: Combining Results with Expert Data

The model above gives valuable insights by simulating election outcomes. It allows to forecast the vote share of a specific party with uncertainty, assigning probabilities to certain events (party A has the highest vote

share, party B gets more share than party A, party A + B have majority in parliament, ..) or by incorporating potential electoral thresholds (vote share for party required to get seats in parliament, e.g. 5% in Germany) one can estimate the seat distribution in the parliament.

However, one might also be interested in how the upcoming government is constituted. For a given election result usually multiple coalitions are conceivable and the government is formed independent from the voters. To answer questions like "How large is the probability that the government consists of a coalition of parties A and B", "What is the probability that person X is going to be chancellor / prime minister" or "How likely is it that party C is part of the government", the incorporation of an expert poll is proposed. Experts are defined as people associated and familiar with politics, e.g. active in politics, working for a party or a politics-related institution or having an academic degree in political science.

Optimally, the expert poll data should be independent (or in statistical terms "orthogonal") to the election results. To achieve that, a list of potential coalitions was given to the experts and the experts are given the task to rank these coalitions under the premise to do this independently to the potential election result. To give a picture, it was stated that the experts should think of a situation, where representatives of all parties are locked into a room with the goal to form coalitions and they should name the parties which would get together first. If that coalition is ruled out, who would find together next and so on. The interviews were assisted to prevent errors and misunderstandings.

How does this help to answer the questions mentioned above? Consider a simplified example with parties A, B and C. The state-space model gave 100 posterior samples of the election outcome and thus one has 100 parliament seat share distributions. These constitute to 4 scenarios:

1. In 10 simulations only coalitions with parties A+B or A+B+C had a majority
2. In 15 simulations A+B or A+C or A+B+C had a majority
3. In 60 simulations A or A+B or A+C or A+B+C had a majority
4. In 25 simulations A+C or A+B+C had a majority

Now, for each scenario the, say 20, expert rankings are considered. The possible coalitions with a majority are selected and it is checked how many expert ranked a coalition first among this selection (which obviously follows a multinomial distribution):

1. 20 / 20 experts ranked A+B above A+B+C, thus A+B gets 100% probability
2. 3 / 20 experts ranked A+B above A+B+C and A+C, 17/20 A+C above A+B and A+B+C, thus A+B gets 15% probability and A+C 85%
3. 20 / 20 experts ranked A above A+B, A+C and A+B+C , thus A gets 100% probability
4. 20 / 20 experts ranked A+C above A+B+C, thus A+C gets 100% probability

Thus, for coalition A+B the total probability is (10 * 100% + 15 * 15%) / 100 = 2.25%. For coalition A+C, it computes to 15 * 85% + 25 * 100%) / 100 = 37.75%, for 'coalition' A (60 * 100%) / 100 = 60% and for coalition A+B+C 0%.

To keep the Bayesian workflow, a dirichlet prior was chosen with all α_k set to 0.5, which changes the results slightly. Another small change to the setting described above was to introduce senior and junior partners, e.g. in the upper example for coalition A+B the expert were given two choices, namely with A as senior partner and B as senior partner, who provide the chancellor in each case. As (at least in Germany) there is an unwritten rule that the party with the most votes provides the chancellor, both coalitions rule out each other and the ranking between them is irrelevant. With this assumption one can also estimate the probability for a certain person being chancellor. In general, as coalition majorities are negatively correlated (e.g. A+B and C+D cannot have a majority at the same time), usually a low double-digit sample of experts already leads to high certainty in probability estimates. Besides the most voted party providing the chancellor, another assumption that is made concerns minority governments, which are ruled out. These are completely unusual in Germany (both on federal and state level), but for other countries with similar election system not unheard of and would require non-trivial model adjustments.

The package includes a function with pre-defined coalitions to compute their probabilities. The big coalition (CDU/CSU + SPD) got by far the highest probability assigned (and eventually became government after

election):

```
fact_coalition_prob <- koalitionDE(dataDE$Koalitionen, modelResults, dataPrep, predDate)
coalition_parties <- c("CDU/CSU - SPD", "SPD - Linke - Grüne", "SPD - FDP-Grüne",
                "CDU/CSU - Grüne", "SPD - Grüne", "SPD - CDU/CSU",
                "CDU/CSU - FDP", "CDU/CSU - FDP - Grüne", "Grüne - SPD",
                "Grüne - SPD - Linke", "Grüne - SPD - FDP", "Grüne - CDU/CSU")
fact_coalition_prob$coalition_parties <- coalition_parties
knitr::kable(fact_coalition_prob[, c(1, 4, 3)])
```

	date_forecast	coalition_parties	estimate
coalition_id_1	2017-09-23	CDU/CSU - SPD	0.598
coalition_id_2	2017-09-23	SPD - Linke - Grüne	0.001
coalition_id_3	2017-09-23	SPD - FDP-Grüne	0.000
coalition_id_4	2017-09-23	CDU/CSU - Grüne	0.015
coalition_id_5	2017-09-23	SPD - Grüne	0.000
coalition_id_6	2017-09-23	SPD - CDU/CSU	0.002
coalition_id_7	2017-09-23	CDU/CSU - FDP	0.074
coalition_id_8	2017-09-23	CDU/CSU - FDP - Grüne	0.310
coalition_id_9	2017-09-23	Grüne - SPD	0.000
coalition_id_10	2017-09-23	Grüne - SPD - Linke	0.000
coalition_id_11	2017-09-23	Grüne - SPD - FDP	0.000
coalition_id_12	2017-09-23	Grüne - CDU/CSU	0.000