SSM Simulation Code

Cole Cappello

06-04-2025

Basic SSM Simulation

The first model we'll simulate from can be written as:

$$y_t = \alpha_t + \epsilon_t \tag{1}$$
$$\alpha_{t+1} = \alpha_t + \nu \tag{2}$$

where $\epsilon_t \sim N(0, 1)$ and $\nu = .5$. We'll suppose we have an initial starting alpha of 1

```r
# Number of states
N <- 50

# Nu
nu <- .5

# Initialize alpha vector
alpha <- rep(NA, 50)
alpha[1] <- 1

# Store true latent states
for(i in 1:(length(alpha)-1)) {
    alpha[i+1] <- alpha[i] + nu
}

# Vector of errors
epsilon <- rnorm(n = 50, mean = 0, sd = 1)

# Simulated observations
y <- alpha + epsilon
```

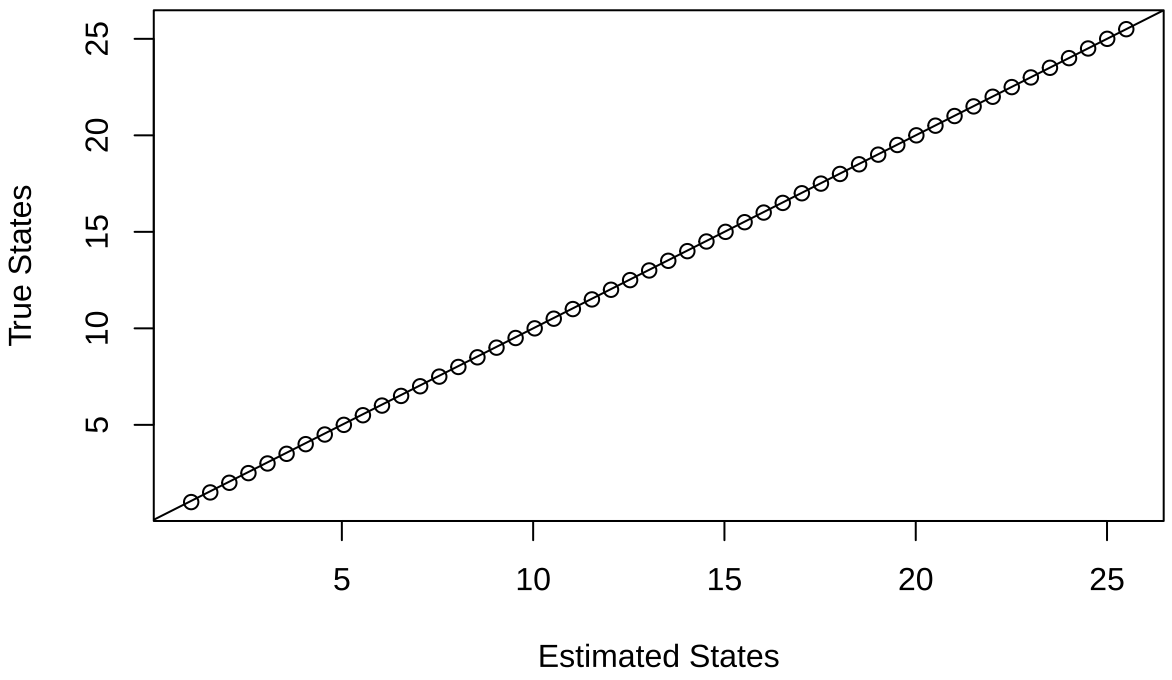
Now we will fit the model with stan and try to recover the parameters. The Linear_SSM.stan file fits a linear SSM with observation error and a deterministic linear process equation.

```r
data_stan <- list(TT = length(y), y = y, z0 = 0)

stan_test <- stan(file = "Linear_SSM.stan",
                  data = data_stan,
                  chains = 3, iter = 3000)
```

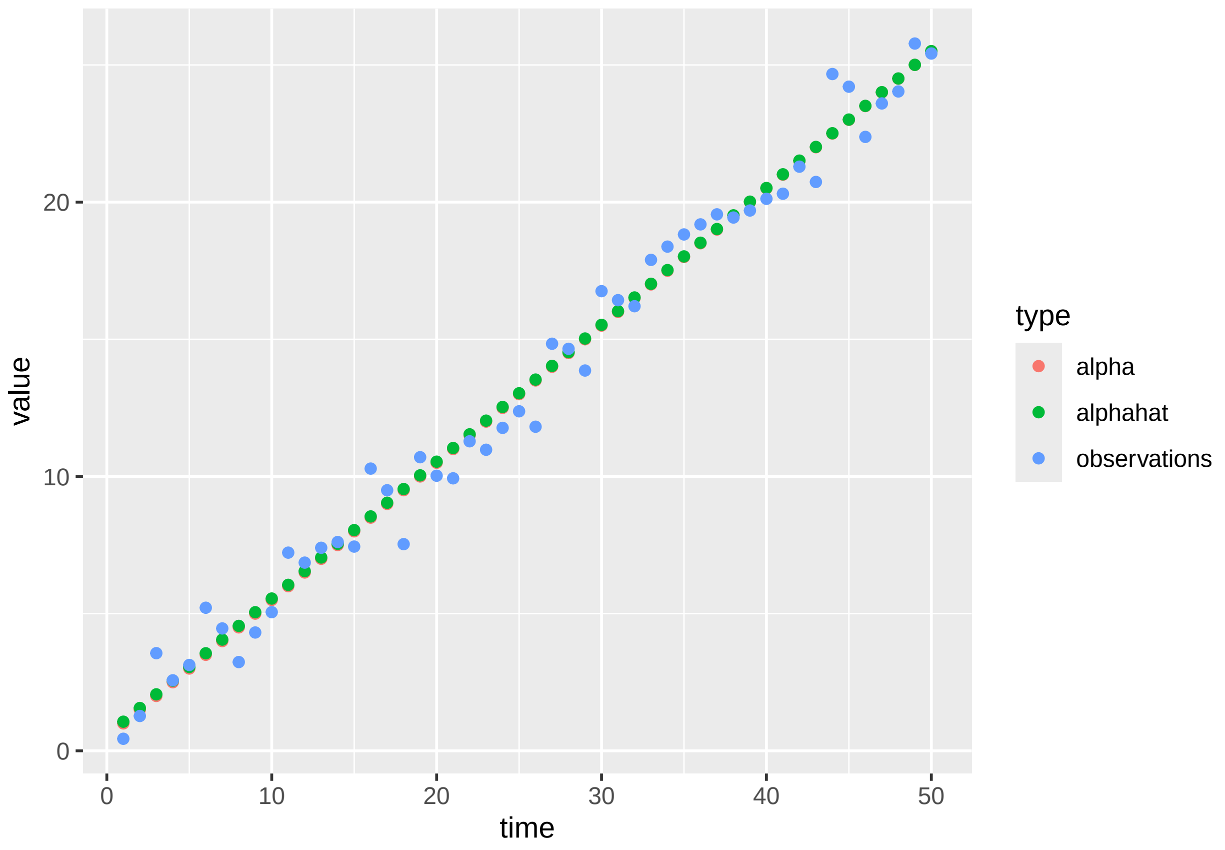
```
## Trying to compile a simple C file

## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## using C compiler: 'Apple clang version 17.0.0 (clang-1700.0.13.3)'
## using SDK: ‘MacOSX15.4.sdk’
## clang -arch arm64 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Frame
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/StanHeade
## In file included from /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/RcppEigen
## In file included from /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/RcppEigen
## /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/RcppEigen/include/Eigen/src/Cor
##   679 | #include <cmath>
##       |          ^~~~~~~
## 1 error generated.
## make: *** [foo.o] Error 1
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 2.2e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.22 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 1: Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 1: Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 1: Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 1: Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 1: Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 1: Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 1: Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 1: Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.04 seconds (Warm-up)
## Chain 1:                0.032 seconds (Sampling)
## Chain 1:                0.072 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.01 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 2: Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 2: Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 2: Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 2: Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 2: Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 2: Iteration: 2400 / 3000 [ 80%]  (Sampling)
```

```
## Chain 2: Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 2: Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.039 seconds (Warm-up)
## Chain 2:                0.026 seconds (Sampling)
## Chain 2:                0.065 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.01 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 3: Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 3: Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 3: Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 3: Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 3: Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 3: Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 3: Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 3: Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.039 seconds (Warm-up)
## Chain 3:                0.029 seconds (Sampling)
## Chain 3:                0.068 seconds (Total)
## Chain 3:
```

```r
states <- c(mean(extract(stan_test, pars = c("z1"))[[1]]), colMeans(extract(stan_test, pars = c("z"))[[

plot(x = states, y = alpha,
     xlab = "Estimated States",
     ylab = "True States")
abline(0,1)
```



```r
# Include a plot of time vs. estimated state, true state, and observations
time <- seq(from = 1, to = 50, by = 1)

df1 <- tibble(time = time,
              alpha = alpha,
              alphahat = states,
              observations = y)


df_long <- df1 %>%
  pivot_longer(cols = c(alpha, alphahat,observations), names_to = "type", values_to = "value")

ggplot(df_long, aes(x = time, y = value, color = type)) +
  geom_point()
```



```
stan_test
```

```
## Inference for Stan model: anon_model.
## 3 chains, each with iter=3000; warmup=1500; thin=1;
## post-warmup draws per chain=1500, total post-warmup draws=4500.
##
##          mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
## sdo      0.95    0.00 0.10   0.79   0.88   0.95   1.01   1.17  2070    1
## z1       1.06    0.01 0.27   0.53   0.88   1.06   1.24   1.60  1599    1
## v        0.50    0.00 0.01   0.48   0.49   0.50   0.51   0.52  1562    1
## z[1]     1.56    0.01 0.26   1.04   1.39   1.56   1.73   2.08  1610    1
## z[2]     2.06    0.01 0.25   1.56   1.89   2.06   2.23   2.56  1623    1
## z[3]     2.56    0.01 0.25   2.07   2.40   2.56   2.72   3.05  1638    1
## z[4]     3.06    0.01 0.24   2.59   2.90   3.06   3.21   3.53  1655    1
## z[5]     3.56    0.01 0.23   3.10   3.40   3.56   3.71   4.02  1675    1
## z[6]     4.05    0.01 0.22   3.62   3.91   4.06   4.20   4.50  1695    1
## z[7]     4.55    0.01 0.22   4.13   4.41   4.56   4.69   4.99  1718    1
## z[8]     5.05    0.00 0.21   4.65   4.92   5.05   5.19   5.47  1745    1
## z[9]     5.55    0.00 0.20   5.16   5.42   5.55   5.68   5.96  1777    1
## z[10]    6.05    0.00 0.19   5.67   5.92   6.05   6.18   6.44  1814    1
## z[11]    6.55    0.00 0.19   6.18   6.43   6.55   6.67   6.92  1859    1
## z[12]    7.05    0.00 0.18   6.69   6.93   7.05   7.16   7.41  1911    1
## z[13]    7.55    0.00 0.17   7.20   7.43   7.55   7.66   7.90  1973    1
## z[14]    8.04    0.00 0.17   7.71   7.93   8.05   8.16   8.38  2047    1
## z[15]    8.54    0.00 0.16   8.22   8.44   8.55   8.65   8.87  2135    1
## z[16]    9.04    0.00 0.16   8.73   8.94   9.04   9.15   9.36  2240    1
## z[17]    9.54    0.00 0.15   9.24   9.44   9.54   9.64   9.85  2365    1
## z[18]  10.04    0.00 0.15   9.74   9.94  10.04  10.14  10.34  2514    1
## z[19]  10.54    0.00 0.15  10.25  10.45  10.54  10.63  10.83  2691    1
## z[20]  11.04    0.00 0.14  10.74  10.95  11.04  11.13  11.32  2898    1
## z[21]  11.54    0.00 0.14  11.25  11.45  11.54  11.63  11.82  3139    1
## z[22]  12.04    0.00 0.14  11.75  11.95  12.04  12.12  12.31  3413    1
## z[23]  12.53    0.00 0.14  12.25  12.45  12.54  12.62  12.81  3716    1
## z[24]  13.03    0.00 0.14  12.75  12.94  13.03  13.12  13.30  4035    1
## z[25]  13.53    0.00 0.14  13.26  13.44  13.53  13.62  13.80  4416    1
## z[26]  14.03    0.00 0.14  13.76  13.94  14.03  14.12  14.30  4774    1
## z[27]  14.53    0.00 0.14  14.26  14.44  14.53  14.62  14.80  5042    1
## z[28]  15.03    0.00 0.14  14.75  14.94  15.03  15.12  15.30  5171    1
## z[29]  15.53    0.00 0.14  15.25  15.44  15.53  15.62  15.80  5168    1
## z[30]  16.03    0.00 0.15  15.74  15.93  16.02  16.12  16.31  5048    1
## z[31]  16.52    0.00 0.15  16.23  16.43  16.52  16.62  16.82  4840    1
## z[32]  17.02    0.00 0.15  16.72  16.92  17.02  17.12  17.33  4500    1
## z[33]  17.52    0.00 0.16  17.21  17.42  17.52  17.63  17.83  4213    1
## z[34]  18.02    0.00 0.16  17.70  17.91  18.02  18.13  18.35  3942    1
## z[35]  18.52    0.00 0.17  18.19  18.41  18.52  18.63  18.85  3695    1
## z[36]  19.02    0.00 0.18  18.68  18.90  19.02  19.14  19.36  3474    1
## z[37]  19.52    0.00 0.18  19.16  19.40  19.51  19.64  19.87  3286    1
## z[38]  20.02    0.00 0.19  19.65  19.89  20.01  20.14  20.38  3133    1
## z[39]  20.52    0.00 0.19  20.14  20.39  20.51  20.64  20.90  2997    1
## z[40]  21.01    0.00 0.20  20.62  20.88  21.01  21.15  21.41  2877    1
## z[41]  21.51    0.00 0.21  21.11  21.37  21.51  21.65  21.92  2771    1
## z[42]  22.01    0.00 0.22  21.59  21.87  22.01  22.15  22.44  2676    1
## z[43]  22.51    0.00 0.22  22.07  22.36  22.51  22.66  22.95  2593    1
## z[44]  23.01    0.00 0.23  22.56  22.85  23.01  23.16  23.46  2518    1
## z[45]  23.51    0.00 0.24  23.04  23.35  23.51  23.67  23.98  2452    1
## z[46]  24.01    0.01 0.25  23.52  23.84  24.00  24.17  24.49  2392    1
## z[47]  24.51    0.01 0.25  24.00  24.33  24.50  24.67  25.01  2339    1
## z[48]  25.00    0.01 0.26  24.49  24.83  25.00  25.18  25.52  2291    1
## z[49]  25.50    0.01 0.27  24.97  25.32  25.50  25.68  26.04  2248    1
## lp__  -22.73    0.03 1.32 -26.26 -23.29 -22.38 -21.78 -21.26  1469    1
##
## Samples were drawn using NUTS(diag_e) at Mon Jun  9 22:50:49 2025.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

We are able to recover the states and a parameters quite well!

Deterministic SSM with an Exponential Increase in the Process Equation

Here we'll simulate from the model

$$y_t = \alpha_t + \epsilon_t \tag{3}$$
$$\alpha_{t+1} = \beta \cdot \alpha_t \tag{4}$$

where $\epsilon_t \sim N(0, 1)$ and $\beta = 1.1$

```r
# Number of states
N <- 50

# Beta
beta <- 1.1

# Initialize alpha vector
alpha <- rep(NA, 50)
alpha[1] <- 1

# Store true latent states
for(i in 1:(length(alpha)-1)) {
    alpha[i+1] <- beta*alpha[i]
}

# Vector of errors
epsilon <- rnorm(n = 50, mean = 0, sd = 5)

# Simulated observations
y <- alpha + epsilon
```

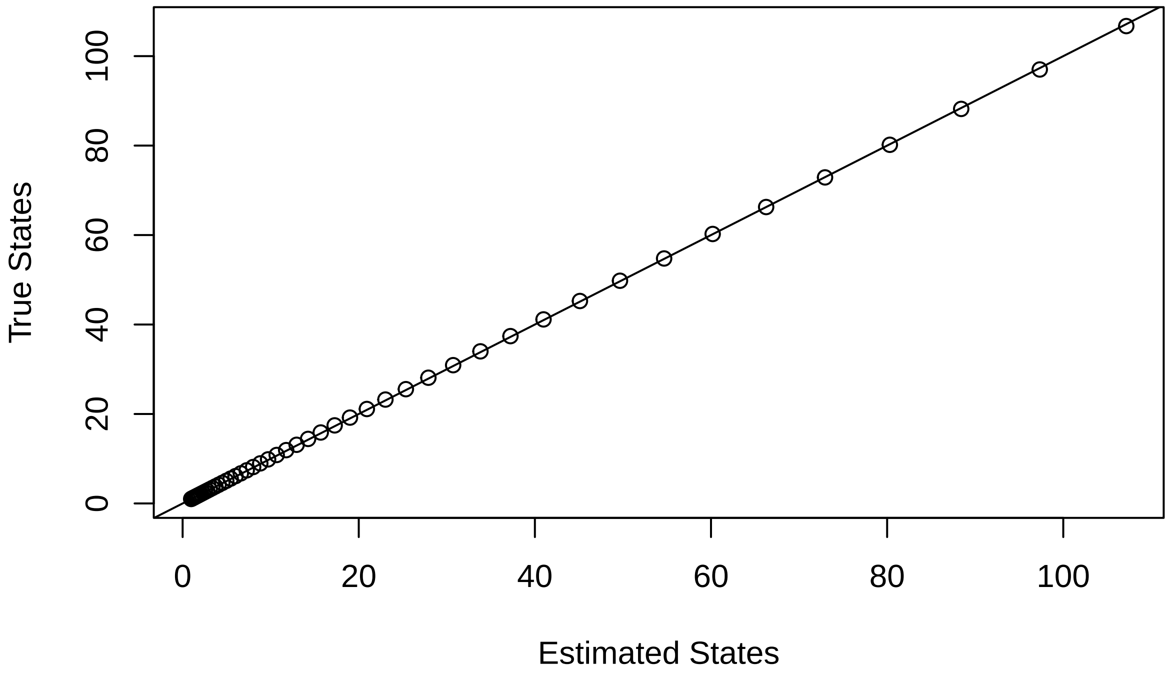
Again we will fit the model with stan and try to recover the parameters. The Basic_SSM.stan file fits a linear SSM with observation error and a deterministic linear process equation.

```r
data_stan <- list(TT = length(y), y = y, z0 = 0)

stan_test <- stan(file = "Basic_SSM.stan",
                  data = data_stan,
                  chains = 3, iter = 3000)
```

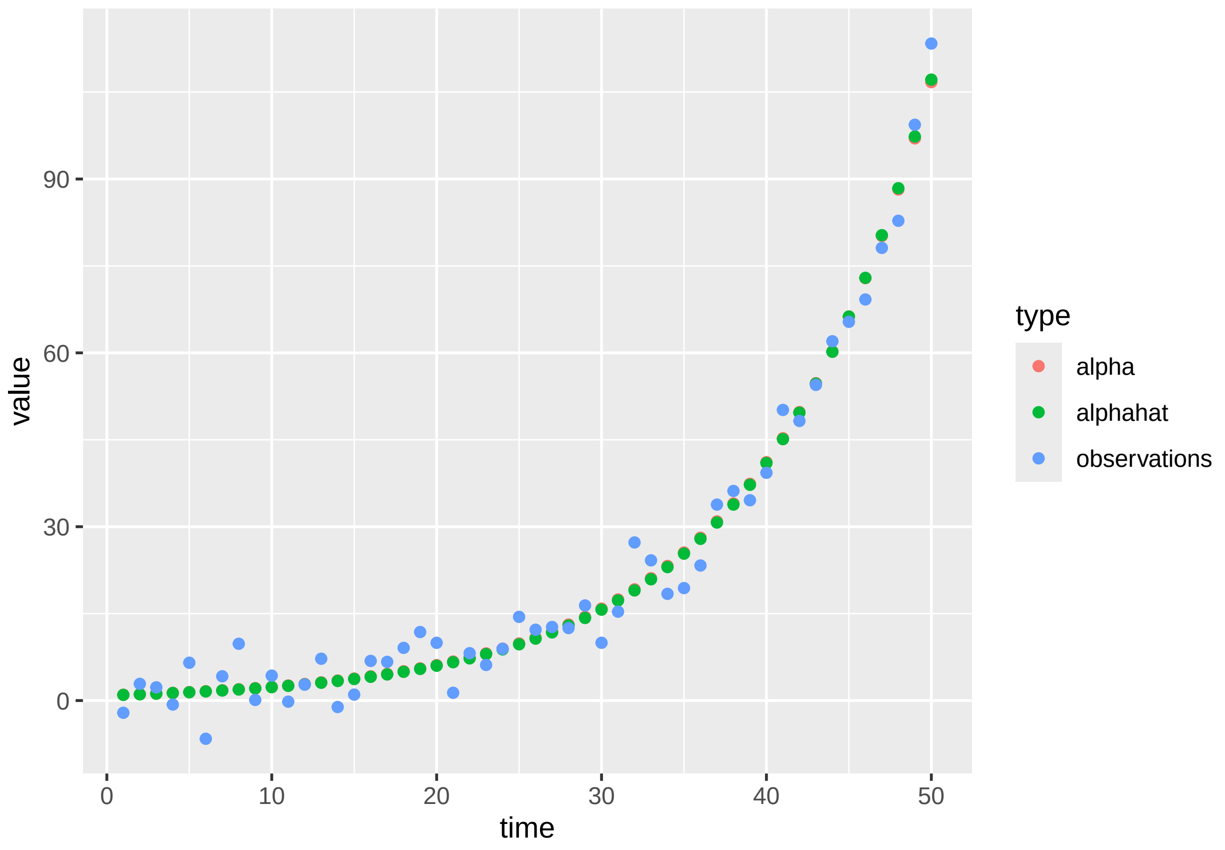
```
## Trying to compile a simple C file

## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## using C compiler: ‘Apple clang version 17.0.0 (clang-1700.0.13.3)’
## using SDK: ‘MacOSX15.4.sdk’
## clang -arch arm64 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Frame
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/StanHeade
## In file included from /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/RcppEigen
## In file included from /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/RcppEigen
## /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/RcppEigen/include/Eigen/src/Cor
##   679 | #include <cmath>
##       |          ^~~~~~~
## 1 error generated.
## make: *** [foo.o] Error 1
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 2.1e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.21 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 1: Iteration:  300 / 3000 [ 10%]  (Warmup)
```

```
## Chain 1: Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 1: Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 1: Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 1: Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 1: Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 1: Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 1: Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.21 seconds (Warm-up)
## Chain 1:                0.111 seconds (Sampling)
## Chain 1:                0.321 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 2e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.02 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 2: Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 2: Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 2: Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 2: Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 2: Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 2: Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 2: Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 2: Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.212 seconds (Warm-up)
## Chain 2:                0.113 seconds (Sampling)
## Chain 2:                0.325 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 2e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.02 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 3: Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 3: Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 3: Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 3: Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 3: Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 3: Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 3: Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 3: Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.361 seconds (Warm-up)
## Chain 3:                0.875 seconds (Sampling)
## Chain 3:                1.236 seconds (Total)
## Chain 3:
```

```
states <- c(mean(extract(stan_test, pars = c("z1"))[[1]]), colMeans(extract(stan_test, pars = c("z"))[[

plot(x = states, y = alpha,
     xlab = "Estimated States",
     ylab = "True States")
abline(0,1)
```



```
# Include a plot of time vs. estimated state, true state, and observations
time <- seq(from = 1, to = 50, by = 1)

df1 <- tibble(time = time,
              alpha = alpha,
              alphahat = states,
              observations = y)


df_long <- df1 %>%
  pivot_longer(cols = c(alpha, alphahat,observations), names_to = "type", values_to = "value")

ggplot(df_long, aes(x = time, y = value, color = type)) +
  geom_point()
```



```
stan_test
```

```
## Inference for Stan model: anon_model.
## 3 chains, each with iter=3000; warmup=1500; thin=1;
## post-warmup draws per chain=1500, total post-warmup draws=4500.
##
##          mean se_mean   sd    2.5%    25%    50%    75%  97.5% n_eff Rhat
## sdo      3.55    0.01 0.31    3.00   3.34   3.54   3.75   4.21  1081    1
## z1       0.97    0.00 0.12    0.76   0.89   0.97   1.05   1.22  1803    1
## beta     1.10    0.00 0.00    1.10   1.10   1.10   1.10   1.11  1904    1
## z[1]     1.07    0.00 0.13    0.84   0.98   1.06   1.15   1.33  1805    1
## z[2]     1.18    0.00 0.14    0.93   1.08   1.17   1.27   1.46  1807    1
## z[3]     1.30    0.00 0.15    1.03   1.19   1.29   1.39   1.60  1809    1
## z[4]     1.43    0.00 0.16    1.14   1.32   1.42   1.53   1.75  1811    1
## z[5]     1.57    0.00 0.17    1.26   1.45   1.56   1.68   1.92  1813    1
## z[6]     1.73    0.00 0.18    1.39   1.60   1.72   1.85   2.10  1815    1
## z[7]     1.90    0.00 0.19    1.54   1.77   1.89   2.03   2.30  1817    1
## z[8]     2.09    0.00 0.21    1.71   1.95   2.09   2.23   2.52  1819    1
## z[9]     2.30    0.01 0.22    1.89   2.15   2.30   2.45   2.76  1821    1
## z[10]    2.53    0.01 0.24    2.09   2.37   2.53   2.69   3.02  1824    1
## z[11]    2.79    0.01 0.25    2.31   2.61   2.78   2.96   3.31  1826    1
## z[12]    3.07    0.01 0.27    2.56   2.88   3.06   3.25   3.62  1828    1
## z[13]    3.38    0.01 0.29    2.84   3.18   3.37   3.57   3.96  1831    1
## z[14]    3.72    0.01 0.31    3.14   3.50   3.71   3.92   4.34  1833    1
## z[15]    4.09    0.01 0.33    3.48   3.87   4.08   4.31   4.75  1836    1
## z[16]    4.51    0.01 0.35    3.85   4.26   4.50   4.74   5.21  1839    1
## z[17]    4.96    0.01 0.37    4.26   4.70   4.95   5.21   5.70  1842    1
## z[18]   5.46    0.01 0.40    4.71   5.19   5.45   5.72   6.25  1845    1
## z[19]   6.01    0.01 0.42    5.21   5.72   6.00   6.29   6.84  1849    1
## z[20]   6.61    0.01 0.45    5.77   6.31   6.60   6.91   7.49  1852    1
## z[21]   7.28    0.01 0.47    6.38   6.96   7.27   7.60   8.21  1856    1
## z[22]   8.01    0.01 0.50    7.06   7.67   8.00   8.35   9.00  1861    1
## z[23]   8.82    0.01 0.52    7.82   8.46   8.81   9.17   9.86  1866    1
## z[24]   9.71    0.01 0.55    8.65   9.33   9.70  10.08  10.80  1871    1
## z[25]  10.68    0.01 0.58    9.57  10.29  10.67  11.07  11.83  1878    1
## z[26]  11.76    0.01 0.61   10.59  11.35  11.75  12.17  12.96  1885    1
## z[27]  12.95    0.01 0.63   11.73  12.52  12.93  13.37  14.20  1893    1
## z[28]  14.25    0.02 0.66   12.98  13.80  14.24  14.69  15.55  1904    1
## z[29]  15.69    0.02 0.69   14.36  15.22  15.68  16.14  17.03  1918    1
## z[30]  17.27    0.02 0.71   15.89  16.79  17.26  17.74  18.66  1935    1
## z[31]  19.01    0.02 0.74   17.58  18.52  19.00  19.49  20.45  1954    1
## z[32]  20.92    0.02 0.76   19.45  20.42  20.92  21.42  22.42  1974    1
## z[33]  23.03    0.02 0.78   21.53  22.51  23.03  23.54  24.57  2000    1
## z[34]  25.35    0.02 0.79   23.82  24.83  25.35  25.86  26.93  2032    1
## z[35]  27.91    0.02 0.81   26.34  27.38  27.90  28.42  29.51  2075    1
## z[36]  30.72    0.02 0.82   29.13  30.19  30.72  31.24  32.36  2132    1
## z[37]  33.82    0.02 0.82   32.21  33.29  33.81  34.34  35.47  2210    1
## z[38]  37.23    0.02 0.83   35.62  36.70  37.22  37.76  38.90  2318    1
## z[39]  40.98    0.02 0.83   39.37  40.46  40.97  41.52  42.65  2487    1
## z[40]  45.11    0.02 0.83   43.49  44.60  45.11  45.64  46.79  2752    1
## z[41]  49.66    0.01 0.83   47.98  49.14  49.66  50.21  51.34  3120    1
## z[42]  54.67    0.01 0.85   52.96  54.13  54.66  55.23  56.38  3605    1
## z[43]  60.19    0.01 0.88   58.43  59.62  60.18  60.78  61.95  4250    1
## z[44]  66.26    0.01 0.94   64.43  65.63  66.27  66.90  68.12  4693    1
## z[45]  72.95    0.01 1.04   70.92  72.24  72.96  73.65  74.98  4825    1
## z[46]  80.31    0.02 1.19   77.95  79.51  80.31  81.12  82.62  4664    1
## z[47]  88.41    0.02 1.40   85.59  87.50  88.42  89.34  91.18  4336    1
## z[48]  97.33    0.03 1.68   94.00  96.24  97.35  98.46 100.67  3725    1
## z[49] 107.16    0.04 2.03  103.12 105.83 107.16 108.48 111.16  3181    1
## lp__  -98.92    0.04 1.33 -102.39 -99.49 -98.56 -97.95 -97.45  1186    1
##
## Samples were drawn using NUTS(diag_e) at Mon Jun  9 22:51:05 2025.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

Once again we do a decent job recovering the states. I did also try a beta of 1.5 and that went horribly because the process blows up towards the end

Linear SSM with Process Error

This time we'll simulate from a model that has process error:

$$y_t = \alpha_t + \epsilon_t \tag{5}$$
$$\alpha_{t+1} = \alpha_t + \nu + \omega_t \tag{6}$$

where $\epsilon_t \sim N(0,1)$, $\omega_t \sim N(0,1)$ and $\nu = 1$. We'll suppose we have an initial starting alpha of 1

```r
# Number of states
N <- 50

# Nu
nu <- 1

# Process errors
omega <- rnorm(n = 50, mean = 0, sd = 1)

# Initialize alpha vector
alpha <- rep(NA, 50)
alpha[1] <- 1

# Store true latent states
for(i in 1:(length(alpha)-1)) {
    alpha[i+1] <- alpha[i] + nu + omega[i+1]
}

# Vector of observation errors
epsilon <- rnorm(n = 50, mean = 0, sd = 1)

# Simulated observations
y <- alpha + epsilon
```

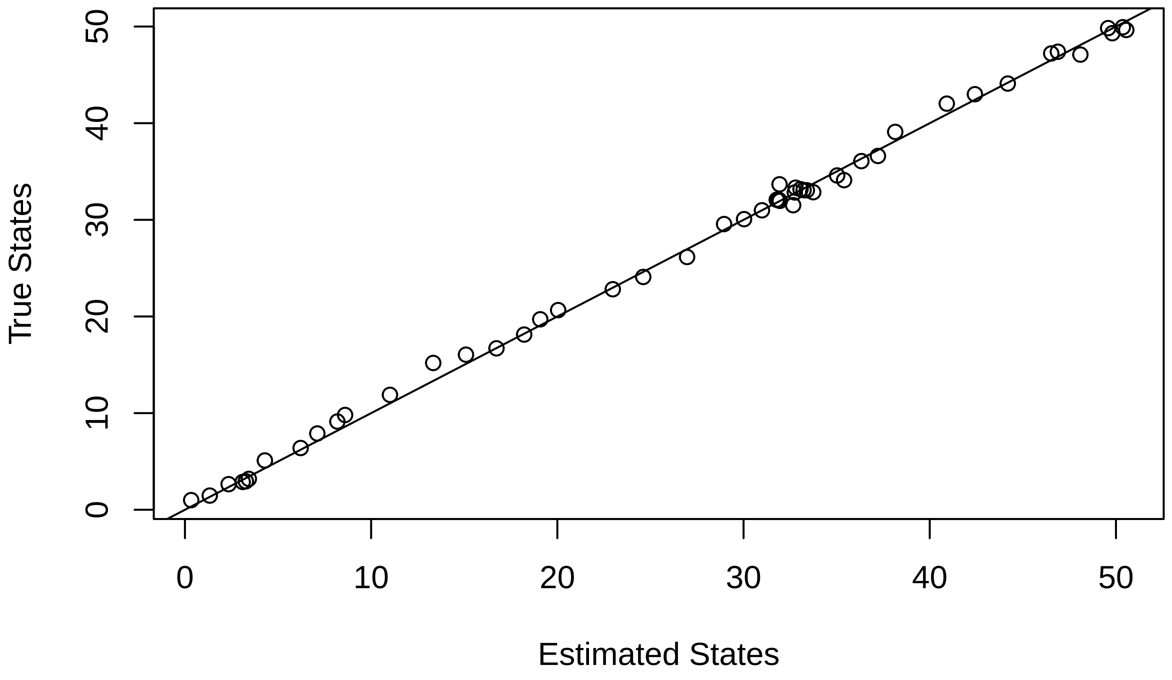
Again we will fit the model with stan and try to recover the parameters. The Basic_SSM2.stan file fits a linear SSM with observation error and process error.

```r
data_stan <- list(TT = length(y), y = y, z0 = 0)

stan_test <- stan(file = "Basic_SSM2.stan",
                  data = data_stan,
                  chains = 3, iter = 3000)
```

```
## Trying to compile a simple C file

## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## using C compiler: ‘Apple clang version 17.0.0 (clang-1700.0.13.3)’
## using SDK: ‘MacOSX15.4.sdk’
## clang -arch arm64 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Frame
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/StanHeade
## In file included from /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/RcppEigen
## In file included from /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/RcppEigen
## /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/RcppEigen/include/Eigen/src/Cor
##   679 | #include <cmath>
##       |          ^~~~~~~
## 1 error generated.
## make: *** [foo.o] Error 1
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 2.3e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.23 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
```

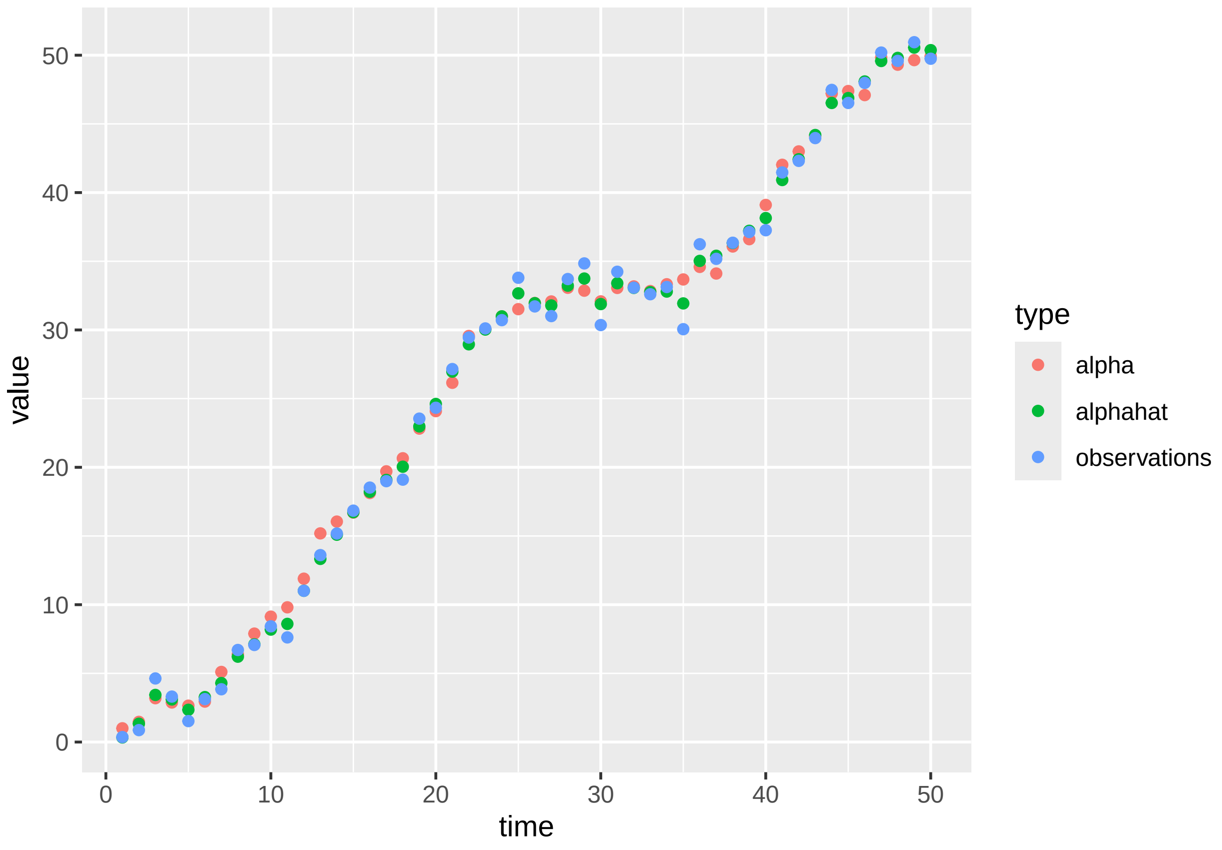
```
## Chain 1:
## Chain 1: Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 1: Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 1: Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 1: Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 1: Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 1: Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 1: Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 1: Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 1: Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.105 seconds (Warm-up)
## Chain 1:                0.09 seconds (Sampling)
## Chain 1:                0.195 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 3e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.03 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 2: Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 2: Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 2: Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 2: Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 2: Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 2: Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 2: Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 2: Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.114 seconds (Warm-up)
## Chain 2:                0.167 seconds (Sampling)
## Chain 2:                0.281 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 3e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.03 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 3000 [  0%]  (Warmup)
## Chain 3: Iteration:  300 / 3000 [ 10%]  (Warmup)
## Chain 3: Iteration:  600 / 3000 [ 20%]  (Warmup)
## Chain 3: Iteration:  900 / 3000 [ 30%]  (Warmup)
## Chain 3: Iteration: 1200 / 3000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1500 / 3000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1501 / 3000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1800 / 3000 [ 60%]  (Sampling)
## Chain 3: Iteration: 2100 / 3000 [ 70%]  (Sampling)
## Chain 3: Iteration: 2400 / 3000 [ 80%]  (Sampling)
## Chain 3: Iteration: 2700 / 3000 [ 90%]  (Sampling)
## Chain 3: Iteration: 3000 / 3000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.102 seconds (Warm-up)
## Chain 3:                0.089 seconds (Sampling)
## Chain 3:                0.191 seconds (Total)
## Chain 3:
```

```
states <- colMeans(extract(stan_test, pars = c("z"))[[1]])

plot(x = states, y = alpha,
     xlab = "Estimated States",
     ylab = "True States")
abline(0,1)
```



```
# Include a plot of time vs. estimated state, true state, and observations
time <- seq(from = 1, to = 50, by = 1)

df1 <- tibble(time = time,
              alpha = alpha,
              alphahat = states,
              observations = y)



df_long <- df1 %>%
  pivot_longer(cols = c(alpha, alphahat,observations), names_to = "type", values_to = "value")
```

```r
ggplot(df_long, aes(x = time, y = value, color = type)) +
  geom_point()
```



```r
stan_test
```

```
## Inference for Stan model: anon_model.
## 3 chains, each with iter=3000; warmup=1500; thin=1;
## post-warmup draws per chain=1500, total post-warmup draws=4500.
##
##          mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
## sdo      1.02    0.01 0.26   0.50   0.86   1.03   1.19   1.51   672 1.01
## sdp      1.46    0.01 0.29   0.96   1.25   1.43   1.65   2.08  1096 1.00
## z[1]     0.34    0.01 0.83  -1.27  -0.21   0.34   0.87   1.97  5896 1.00
## z[2]     1.33    0.01 0.80  -0.22   0.82   1.32   1.85   2.99  5261 1.00
## z[3]     3.43    0.02 0.86   1.68   2.86   3.47   4.03   5.00  1776 1.00
## z[4]     3.10    0.01 0.76   1.59   2.60   3.12   3.60   4.55  4892 1.00
## z[5]     2.35    0.02 0.84   0.80   1.77   2.32   2.90   4.07  2200 1.00
## z[6]     3.27    0.01 0.77   1.81   2.78   3.25   3.76   4.80  4333 1.00
## z[7]     4.29    0.01 0.80   2.68   3.77   4.27   4.80   5.91  3879 1.00
## z[8]     6.21    0.01 0.77   4.64   5.71   6.24   6.73   7.65  4362 1.00
## z[9]     7.11    0.01 0.75   5.64   6.61   7.13   7.61   8.58  5838 1.00
## z[10]    8.19    0.01 0.75   6.74   7.69   8.19   8.69   9.65  6316 1.00
## z[11]    8.60    0.02 0.84   7.03   8.01   8.57   9.14  10.36  2187 1.00
## z[12]   11.01    0.01 0.76   9.49  10.52  11.02  11.51  12.48  4798 1.00
## z[13]   13.33    0.01 0.78  11.75  12.83  13.33  13.85  14.89  5355 1.00
## z[14]   15.09    0.01 0.76  13.53  14.60  15.12  15.59  16.57  4971 1.00
## z[15]   16.73    0.01 0.78  15.14  16.24  16.75  17.24  18.20  4942 1.00
## z[16]   18.22    0.01 0.76  16.69  17.73  18.24  18.73  19.66  4254 1.00
## z[17]   19.08    0.01 0.79  17.54  18.55  19.07  19.61  20.70  4573 1.00
## z[18]   20.04    0.02 0.84  18.51  19.47  20.01  20.60  21.76  2654 1.00
## z[19]   22.98    0.01 0.77  21.43  22.48  22.99  23.50  24.43  4619 1.00
## z[20]   24.61    0.01 0.77  23.16  24.12  24.60  25.10  26.14  5416 1.00
## z[21]   26.97    0.01 0.78  25.40  26.46  26.98  27.47  28.48  4527 1.00
## z[22]   28.95    0.01 0.81  27.31  28.43  28.98  29.49  30.51  3676 1.00
## z[23]   30.03    0.01 0.78  28.49  29.52  30.03  30.53  31.52  5208 1.00
## z[24]   30.99    0.01 0.75  29.45  30.52  31.00  31.47  32.46  4403 1.00
## z[25]   32.66    0.02 0.88  30.87  32.08  32.70  33.27  34.26  2193 1.00
## z[26]   31.95    0.01 0.77  30.47  31.43  31.95  32.48  33.46  5102 1.00
## z[27]   31.78    0.01 0.79  30.34  31.22  31.76  32.30  33.41  3517 1.00
## z[28]   33.22    0.01 0.80  31.62  32.70  33.25  33.76  34.74  3861 1.00
## z[29]   33.74    0.02 0.85  32.08  33.17  33.76  34.32  35.33  2400 1.00
## z[30]   31.88    0.02 0.89  30.20  31.26  31.88  32.49  33.63  1827 1.00
## z[31]   33.40    0.01 0.80  31.80  32.89  33.42  33.93  34.92  3702 1.00
## z[32]   33.06    0.01 0.78  31.52  32.55  33.06  33.57  34.62  5926 1.00
## z[33]   32.75    0.01 0.77  31.25  32.25  32.75  33.23  34.30  5311 1.00
## z[34]   32.79    0.01 0.77  31.28  32.29  32.79  33.30  34.35  5404 1.00
## z[35]   31.93    0.03 0.98  30.10  31.23  31.92  32.59  33.93  1388 1.00
## z[36]   35.03    0.02 0.84  33.30  34.48  35.04  35.62  36.62  2647 1.00
## z[37]   35.40    0.01 0.75  33.91  34.93  35.39  35.88  36.91  5509 1.00
## z[38]   36.33    0.01 0.77  34.82  35.80  36.32  36.84  37.84  6256 1.00
## z[39]   37.22    0.01 0.77  35.71  36.70  37.21  37.72  38.78  5263 1.00
## z[40]   38.15    0.02 0.87  36.52  37.55  38.12  38.70  39.96  2784 1.00
## z[41]   40.91    0.01 0.79  39.34  40.40  40.93  41.44  42.42  3937 1.00
## z[42]   42.43    0.01 0.78  40.88  41.92  42.42  42.95  43.99  5233 1.00
## z[43]   44.19    0.01 0.77  42.68  43.71  44.19  44.69  45.70  5186 1.00
## z[44]   46.52    0.02 0.83  44.87  45.95  46.55  47.09  48.07  2563 1.00
## z[45]   46.88    0.01 0.76  45.38  46.36  46.88  47.39  48.40  5337 1.00
## z[46]   48.09    0.01 0.74  46.63  47.59  48.07  48.59  49.57  5296 1.00
## z[47]   49.58    0.01 0.80  48.01  49.05  49.60  50.13  51.14  4142 1.00
## z[48]   49.81    0.01 0.77  48.31  49.31  49.79  50.30  51.30  5640 1.00
## z[49]   50.55    0.01 0.79  48.98  50.03  50.56  51.07  52.08  5531 1.00
## z[50]   50.37    0.01 0.92  48.67  49.76  50.31  50.94  52.31  3878 1.00
## v        1.02    0.00 0.22   0.60   0.88   1.02   1.16   1.46  6568 1.00
## lp__   -65.44    0.38 9.44 -83.62 -71.27 -65.67 -60.08 -44.90   622 1.01
##
## Samples were drawn using NUTS(diag_e) at Mon Jun  9 22:51:21 2025.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

Here we see things start to break down a bit. It might be interesting now to compare the mean squared error of our estimates for the latent states versus the observations themselves to makes sure we are still doing better.

```r
calc_rMSE <- function(alpha,estimate) {
  n <- length(alpha)
  rMSE <- 0
  for(i in 1:n){
    rMSE <- rMSE + (estimate[i] - alpha[i])^2
  }
  rMSE <- sqrt(rMSE/n)
  return(rMSE)
}

calc_rMSE(alpha,states)
```

```
## [1] 0.7099657
```

```r
calc_rMSE(alpha, y)
```

```
## [1] 1.105294
```

Still doing better than the observations. What happens though if we increase the standard errors? lets do the same model with $\epsilon_t \sim N(0,2)$, $\omega_t \sim N(0,2)$ and $\nu = 1$

```r
# Number of states
N <- 50

# Nu
nu <- 1

# Process errors
omega <- rnorm(n = 50, mean = 0, sd = 2)

# Initialize alpha vector
alpha <- rep(NA, 50)
alpha[1] <- 1

# Store true latent states
for(i in 1:(length(alpha)-1)) {
    alpha[i+1] <- alpha[i] + nu + omega[i+1]
}

# Vector of observation errors
epsilon <- rnorm(n = 50, mean = 0, sd = 2)

# Simulated observations
y <- alpha + epsilon
```

Again we will fit the model with stan and try to recover the parameters. The Basic_SSM2.stan file fits a linear SSM with observation error and process error.

```r
data_stan <- list(TT = length(y), y = y, z0 = 0)

stan_test <- stan(file = "Basic_SSM2.stan",
                  data = data_stan,
                  chains = 3, iter = 5000)
```

```
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 5e-06 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.05 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 5000 [  0%]  (Warmup)
## Chain 1: Iteration:  500 / 5000 [ 10%]  (Warmup)
## Chain 1: Iteration: 1000 / 5000 [ 20%]  (Warmup)
## Chain 1: Iteration: 1500 / 5000 [ 30%]  (Warmup)
## Chain 1: Iteration: 2000 / 5000 [ 40%]  (Warmup)
## Chain 1: Iteration: 2500 / 5000 [ 50%]  (Warmup)
## Chain 1: Iteration: 2501 / 5000 [ 50%]  (Sampling)
## Chain 1: Iteration: 3000 / 5000 [ 60%]  (Sampling)
## Chain 1: Iteration: 3500 / 5000 [ 70%]  (Sampling)
## Chain 1: Iteration: 4000 / 5000 [ 80%]  (Sampling)
## Chain 1: Iteration: 4500 / 5000 [ 90%]  (Sampling)
## Chain 1: Iteration: 5000 / 5000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.203 seconds (Warm-up)
## Chain 1:                0.215 seconds (Sampling)
## Chain 1:                0.418 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 3e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.03 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 5000 [  0%]  (Warmup)
## Chain 2: Iteration:  500 / 5000 [ 10%]  (Warmup)
## Chain 2: Iteration: 1000 / 5000 [ 20%]  (Warmup)
## Chain 2: Iteration: 1500 / 5000 [ 30%]  (Warmup)
## Chain 2: Iteration: 2000 / 5000 [ 40%]  (Warmup)
## Chain 2: Iteration: 2500 / 5000 [ 50%]  (Warmup)
## Chain 2: Iteration: 2501 / 5000 [ 50%]  (Sampling)
## Chain 2: Iteration: 3000 / 5000 [ 60%]  (Sampling)
## Chain 2: Iteration: 3500 / 5000 [ 70%]  (Sampling)
## Chain 2: Iteration: 4000 / 5000 [ 80%]  (Sampling)
## Chain 2: Iteration: 4500 / 5000 [ 90%]  (Sampling)
## Chain 2: Iteration: 5000 / 5000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.202 seconds (Warm-up)
## Chain 2:                0.153 seconds (Sampling)
## Chain 2:                0.355 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 3e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.03 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 5000 [  0%]  (Warmup)
## Chain 3: Iteration:  500 / 5000 [ 10%]  (Warmup)
## Chain 3: Iteration: 1000 / 5000 [ 20%]  (Warmup)
## Chain 3: Iteration: 1500 / 5000 [ 30%]  (Warmup)
## Chain 3: Iteration: 2000 / 5000 [ 40%]  (Warmup)
## Chain 3: Iteration: 2500 / 5000 [ 50%]  (Warmup)
## Chain 3: Iteration: 2501 / 5000 [ 50%]  (Sampling)
## Chain 3: Iteration: 3000 / 5000 [ 60%]  (Sampling)
## Chain 3: Iteration: 3500 / 5000 [ 70%]  (Sampling)
## Chain 3: Iteration: 4000 / 5000 [ 80%]  (Sampling)
## Chain 3: Iteration: 4500 / 5000 [ 90%]  (Sampling)
## Chain 3: Iteration: 5000 / 5000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.164 seconds (Warm-up)
## Chain 3:                0.239 seconds (Sampling)
## Chain 3:                0.403 seconds (Total)
## Chain 3:
```

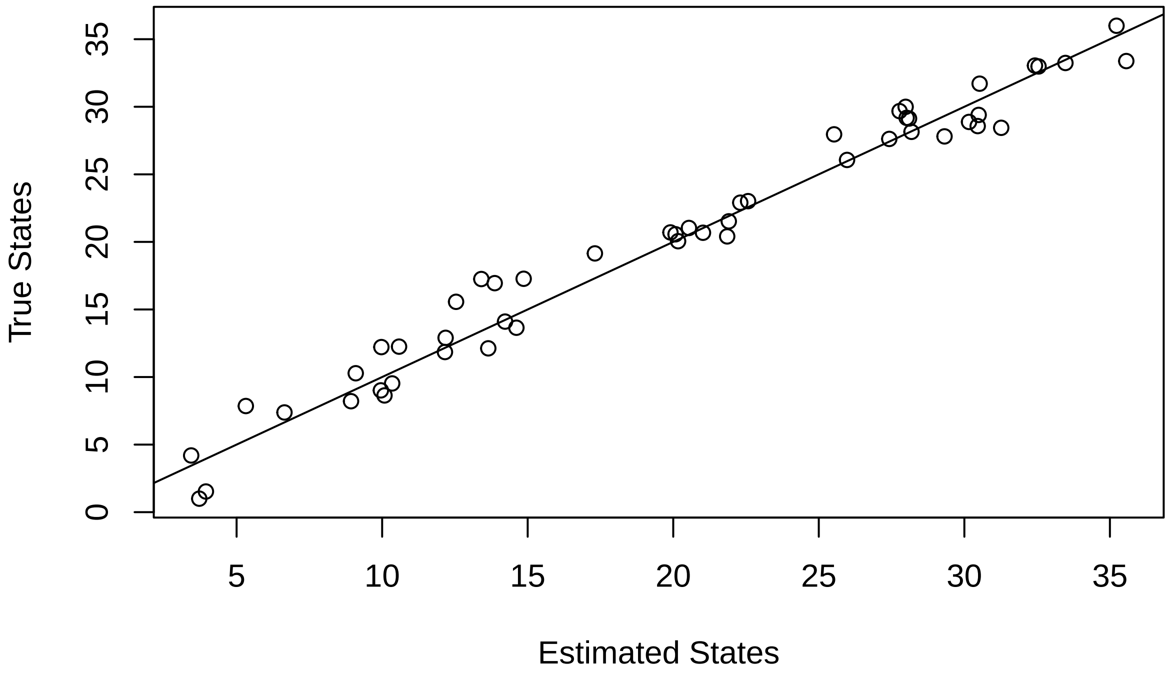
```
## Warning: There were 2 chains where the estimated Bayesian Fraction of Missing Information was low. S
## https://mc-stan.org/misc/warnings.html#bfmi-low
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

```
## Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior means and medians may be
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#bulk-ess
```

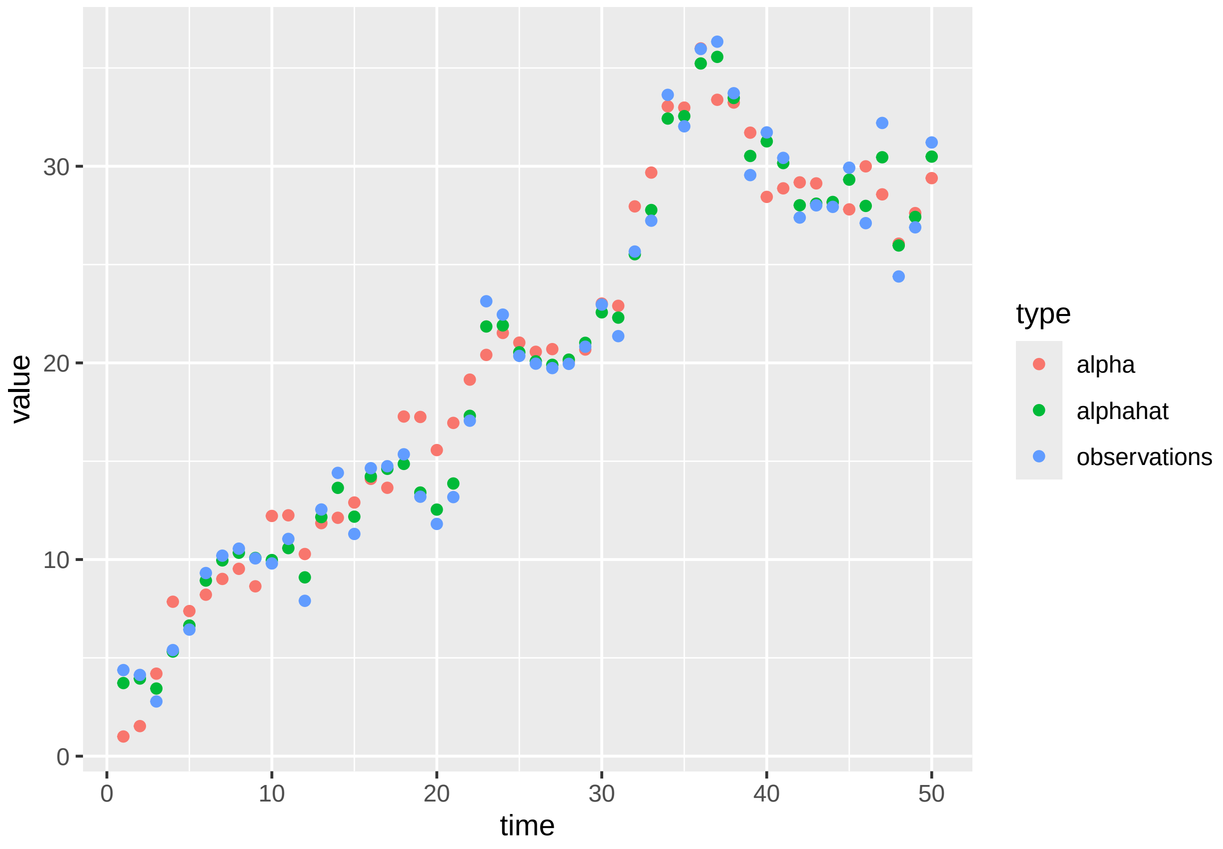
```
## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quant
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#tail-ess
```

```r
states <- colMeans(extract(stan_test, pars = c("z"))[[1]])

plot(x = states, y = alpha,
     xlab = "Estimated States",
     ylab = "True States")
abline(0,1)
```



```r
# Include a plot of time vs. estimated state, true state, and observations
time <- seq(from = 1, to = 50, by = 1)
```

```
df1 <- tibble(time = time,
              alpha = alpha,
              alphahat = states,
              observations = y)


df_long <- df1 %>%
  pivot_longer(cols = c(alpha, alphahat,observations), names_to = "type", values_to = "value")

ggplot(df_long, aes(x = time, y = value, color = type)) +
  geom_point()
```



```
stan_test
```

```
## Inference for Stan model: anon_model.
## 3 chains, each with iter=5000; warmup=2500; thin=1;
## post-warmup draws per chain=2500, total post-warmup draws=7500.
##
##          mean se_mean   sd   2.5%    25%   50%   75%  97.5% n_eff Rhat
## sdo      1.17    0.03 0.46   0.31   0.84  1.20  1.50   2.02   253 1.01
## sdp      2.29    0.02 0.43   1.49   1.99  2.28  2.59   3.17   538 1.00
## z[1]     3.72    0.02 1.07   1.35   3.07  3.84  4.42   5.65  2239 1.00
## z[2]     3.95    0.01 0.98   1.86   3.36  3.98  4.54   5.89  8179 1.00
## z[3]     3.44    0.02 1.04   1.57   2.74  3.33  4.09   5.68  2064 1.00
## z[4]     5.32    0.01 0.94   3.43   4.76  5.32  5.89   7.21  9184 1.00
## z[5]     6.64    0.01 0.98   4.71   6.04  6.60  7.23   8.70  6796 1.00
## z[6]     8.93    0.02 0.99   6.76   8.33  9.01  9.56  10.83  3998 1.00
## z[7]    9.96    0.01  0.99    7.80    9.38  10.01  10.57  11.90  6221 1.00
## z[8]   10.34    0.01  0.99    8.24    9.77  10.40  10.96  12.25  7303 1.00
## z[9]   10.08    0.01  0.95    8.14    9.49  10.08  10.69  12.04  9478 1.00
## z[10]   9.97    0.01  0.96    8.09    9.40   9.94  10.55  11.95  9950 1.00
## z[11]  10.58    0.01  0.97    8.51    9.99  10.66  11.20  12.39  4751 1.00
## z[12]   9.09    0.04  1.17    7.20    8.21   8.96   9.86  11.65   821 1.00
## z[13]  12.16    0.01  0.96   10.13   11.57  12.23  12.75  14.01  5411 1.00
## z[14]  13.65    0.03  1.05   11.32   12.99  13.76  14.36  15.51  1649 1.00
## z[15]  12.18    0.03  1.04   10.37   11.45  12.07  12.84  14.44  1287 1.00
## z[16]  14.22    0.01  0.98   12.12   13.64  14.31  14.83  16.07  4522 1.00
## z[17]  14.61    0.01  0.96   12.54   14.05  14.65  15.20  16.51  9555 1.00
## z[18]  14.86    0.02  0.99   12.73   14.27  14.96  15.50  16.69  3541 1.00
## z[19]  13.41    0.01  0.97   11.53   12.81  13.35  13.97  15.50  8137 1.00
## z[20]  12.54    0.03  1.11   10.63   11.78  12.40  13.23  15.02  1637 1.00
## z[21]  13.87    0.03  1.08   12.01   13.12  13.72  14.50  16.28  1619 1.00
## z[22]  17.31    0.01  0.99   15.35   16.71  17.26  17.91  19.40  8074 1.00
## z[23]  21.85    0.05  1.27   19.06   21.05  22.03  22.83  23.81   676 1.00
## z[24]  21.91    0.02  1.06   19.53   21.27  22.05  22.62  23.78  1944 1.00
## z[25]  20.54    0.01  0.95   18.64   19.96  20.50  21.12  22.50  9047 1.00
## z[26]  20.08    0.01  0.94   18.20   19.50  20.06  20.67  22.02  9702 1.00
## z[27]  19.90    0.01  0.97   18.01   19.31  19.86  20.48  21.94  7180 1.00
## z[28]  20.17    0.01  0.99   18.30   19.55  20.10  20.75  22.28  5976 1.00
## z[29]  21.02    0.01  0.98   19.08   20.42  20.97  21.60  23.10  7380 1.00
## z[30]  22.57    0.01  1.00   20.49   21.96  22.63  23.20  24.51  6880 1.00
## z[31]  22.30    0.03  1.11   20.41   21.51  22.16  23.01  24.74  1278 1.00
## z[32]  25.53    0.01  0.98   23.50   24.96  25.54  26.08  27.50 11503 1.00
## z[33]  27.77    0.02  0.98   25.95   27.13  27.70  28.39  29.84  3475 1.00
## z[34]  32.43    0.04  1.19   29.83   31.68  32.57  33.32  34.33   785 1.00
## z[35]  32.55    0.01  0.98   30.70   31.91  32.47  33.16  34.63  4725 1.00
## z[36]  35.23    0.03  1.08   32.84   34.57  35.37  35.97  37.06  1476 1.00
## z[37]  35.56    0.03  1.10   33.12   34.89  35.70  36.33  37.44  1446 1.00
## z[38]  33.47    0.01  0.96   31.41   32.90  33.53  34.06  35.29  6492 1.00
## z[39]  30.53    0.03  1.10   28.69   29.73  30.41  31.21  32.93  1101 1.00
## z[40]  31.27    0.01  1.00   29.17   30.64  31.34  31.89  33.21  5265 1.00
## z[41]  30.16    0.01  0.97   28.09   29.59  30.21  30.74  32.08  7675 1.00
## z[42]  28.02    0.02  1.03   26.17   27.33  27.90  28.62  30.30  1812 1.00
## z[43]  28.10    0.01  0.97   26.18   27.51  28.08  28.68  30.13  8866 1.00
## z[44]  28.18    0.01  0.96   26.28   27.61  28.13  28.76  30.25  8165 1.00
## z[45]  29.32    0.02  1.01   27.17   28.70  29.43  29.99  31.11  2330 1.00
## z[46]  27.98    0.02  1.03   26.20   27.26  27.88  28.64  30.17  1692 1.00
## z[47]  30.46    0.06  1.33   27.62   29.57  30.56  31.52  32.58   525 1.00
## z[48]  25.97    0.05  1.30   23.92   24.95  25.85  26.84  28.82   563 1.00
## z[49]  27.42    0.02  1.03   25.55   26.75  27.31  28.04  29.67  2530 1.00
## z[50]  30.49    0.02  1.15   27.93   29.78  30.64  31.26  32.52  2263 1.00
## v       0.54    0.01  0.33   -0.11    0.33   0.54   0.75   1.21  3928 1.00
## lp__  -91.96    1.34 18.11 -115.82 -103.79 -96.41 -85.45 -42.12   183 1.02
##
## Samples were drawn using NUTS(diag_e) at Mon Jun  9 22:51:23 2025.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

Hmm, doesn't look as good. What's the rMSE?

```r
# Estimates
calc_rMSE(alpha = alpha, estimate = states)
```

```
## [1] 1.579932
```

```r
# Observations
calc_rMSE(alpha = alpha, estimate = y)
```

```
## [1] 1.986003
```

Not great but we are still doing better than just looking at the observations. What happens if we increase the process error and decrease the observation error?

```r
# Number of states
N <- 50

# Nu
nu <- 1

# Process errors
omega <- rnorm(n = 50, mean = 0, sd = 3)

# Initialize alpha vector
alpha <- rep(NA, 50)
alpha[1] <- 1

# Store true latent states
for(i in 1:(length(alpha)-1)) {
    alpha[i+1] <- alpha[i] + nu + omega[i+1]
}

# Vector of observation errors
epsilon <- rnorm(n = 50, mean = 0, sd = 1)

# Simulated observations
y <- alpha + epsilon
```

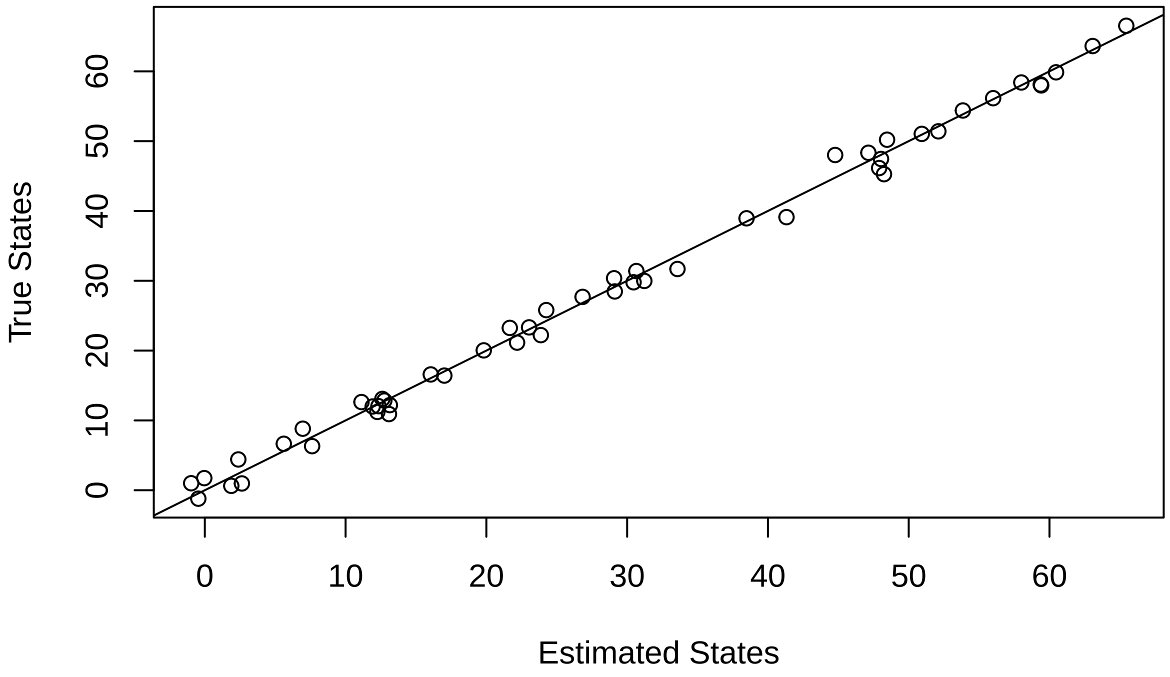
Again we will fit the model with stan and try to recover the parameters. The Basic_SSM2.stan file fits a linear SSM with observation error and process error.

```r
data_stan <- list(TT = length(y), y = y, z0 = 0)

stan_test <- stan(file = "Basic_SSM2.stan",
                  data = data_stan,
                  chains = 3, iter = 15000)
```

```
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 5e-06 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.05 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:     1 / 15000 [  0%]  (Warmup)
## Chain 1: Iteration:  1500 / 15000 [ 10%]  (Warmup)
## Chain 1: Iteration:  3000 / 15000 [ 20%]  (Warmup)
## Chain 1: Iteration:  4500 / 15000 [ 30%]  (Warmup)
## Chain 1: Iteration:  6000 / 15000 [ 40%]  (Warmup)
## Chain 1: Iteration:  7500 / 15000 [ 50%]  (Warmup)
## Chain 1: Iteration:  7501 / 15000 [ 50%]  (Sampling)
## Chain 1: Iteration:  9000 / 15000 [ 60%]  (Sampling)
## Chain 1: Iteration: 10500 / 15000 [ 70%]  (Sampling)
## Chain 1: Iteration: 12000 / 15000 [ 80%]  (Sampling)
## Chain 1: Iteration: 13500 / 15000 [ 90%]  (Sampling)
## Chain 1: Iteration: 15000 / 15000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.631 seconds (Warm-up)
## Chain 1:                0.503 seconds (Sampling)
## Chain 1:                1.134 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 3e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.03 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:     1 / 15000 [  0%]  (Warmup)
## Chain 2: Iteration:  1500 / 15000 [ 10%]  (Warmup)
## Chain 2: Iteration:  3000 / 15000 [ 20%]  (Warmup)
## Chain 2: Iteration:  4500 / 15000 [ 30%]  (Warmup)
## Chain 2: Iteration:  6000 / 15000 [ 40%]  (Warmup)
## Chain 2: Iteration:  7500 / 15000 [ 50%]  (Warmup)
## Chain 2: Iteration:  7501 / 15000 [ 50%]  (Sampling)
## Chain 2: Iteration:  9000 / 15000 [ 60%]  (Sampling)
## Chain 2: Iteration: 10500 / 15000 [ 70%]  (Sampling)
## Chain 2: Iteration: 12000 / 15000 [ 80%]  (Sampling)
## Chain 2: Iteration: 13500 / 15000 [ 90%]  (Sampling)
## Chain 2: Iteration: 15000 / 15000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.482 seconds (Warm-up)
## Chain 2:                0.586 seconds (Sampling)
## Chain 2:                1.068 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 3e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.03 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:     1 / 15000 [  0%]  (Warmup)
## Chain 3: Iteration:  1500 / 15000 [ 10%]  (Warmup)
## Chain 3: Iteration:  3000 / 15000 [ 20%]  (Warmup)
## Chain 3: Iteration:  4500 / 15000 [ 30%]  (Warmup)
## Chain 3: Iteration:  6000 / 15000 [ 40%]  (Warmup)
## Chain 3: Iteration:  7500 / 15000 [ 50%]  (Warmup)
## Chain 3: Iteration:  7501 / 15000 [ 50%]  (Sampling)
## Chain 3: Iteration:  9000 / 15000 [ 60%]  (Sampling)
## Chain 3: Iteration: 10500 / 15000 [ 70%]  (Sampling)
## Chain 3: Iteration: 12000 / 15000 [ 80%]  (Sampling)
## Chain 3: Iteration: 13500 / 15000 [ 90%]  (Sampling)
## Chain 3: Iteration: 15000 / 15000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.502 seconds (Warm-up)
## Chain 3:                0.508 seconds (Sampling)
## Chain 3:                1.01 seconds (Total)
## Chain 3:
```

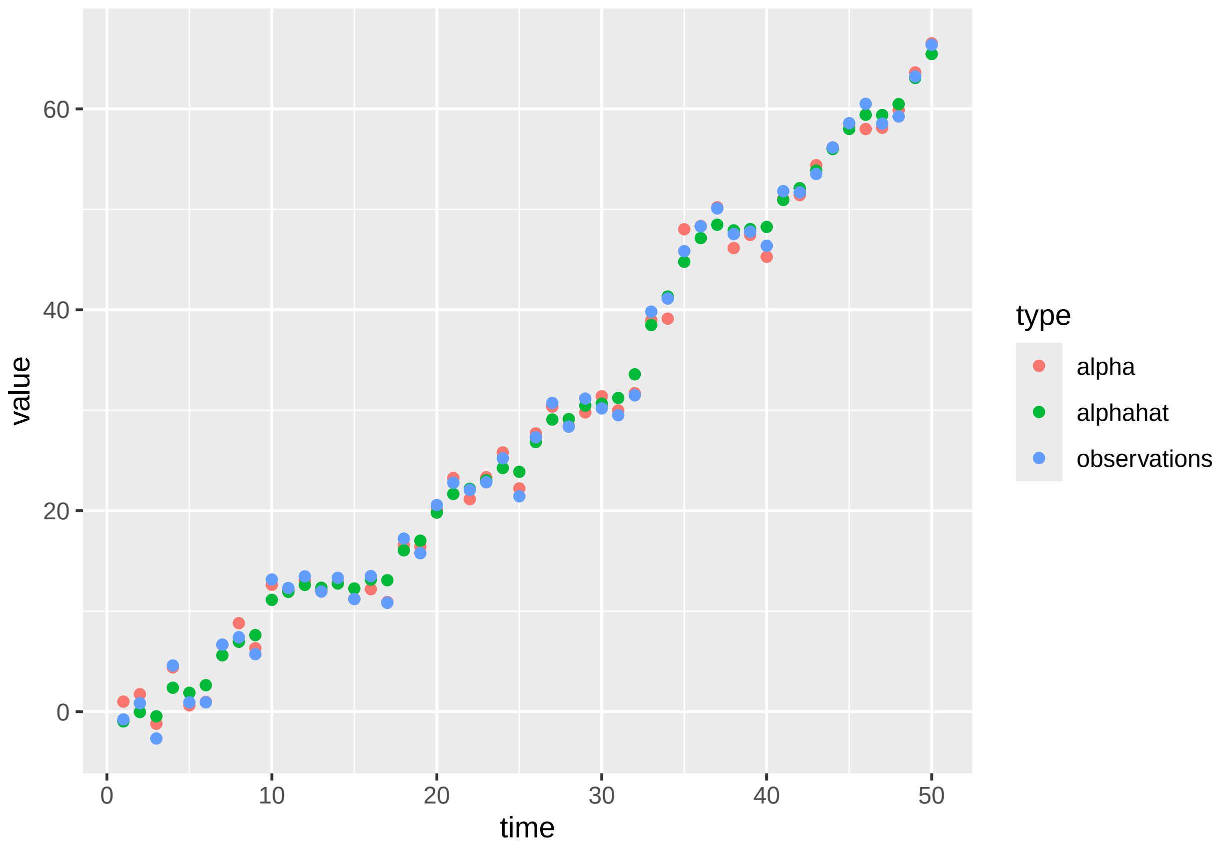
```
## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quant
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#tail-ess
```

```
states <- colMeans(extract(stan_test, pars = c("z"))[[1]])

plot(x = states, y = alpha,
     xlab = "Estimated States",
     ylab = "True States")
abline(0,1)
```



```
# Include a plot of time vs. estimated state, true state, and observations
time <- seq(from = 1, to = 50, by = 1)

df1 <- tibble(time = time,
              alpha = alpha,
              alphahat = states,
              observations = y)


df_long <- df1 %>%
  pivot_longer(cols = c(alpha, alphahat,observations), names_to = "type", values_to = "value")
```

```r
ggplot(df_long, aes(x = time, y = value, color = type)) +
  geom_point()
```



```r
stan_test
```

```
## Inference for Stan model: anon_model.
## 3 chains, each with iter=15000; warmup=7500; thin=1;
## post-warmup draws per chain=7500, total post-warmup draws=22500.
##
##           mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
## sdo       1.69    0.02 0.48   0.50   1.43   1.72   2.01   2.56   492 1.01
## sdp       1.88    0.02 0.54   0.97   1.50   1.82   2.22   3.06   802 1.01
## z[1]     -0.97    0.01 1.23  -3.46  -1.76  -0.94  -0.17   1.46 18454 1.00
## z[2]     -0.04    0.01 1.18  -2.42  -0.81  -0.01   0.77   2.21 10442 1.00
## z[3]     -0.46    0.03 1.33  -2.93  -1.40  -0.45   0.45   2.17  1562 1.00
## z[4]      2.38    0.03 1.31  -0.17   1.47   2.36   3.28   4.84  1645 1.00
## z[5]      1.88    0.02 1.20  -0.38   1.04   1.84   2.68   4.32  6167 1.00
## z[6]      2.63    0.03 1.31   0.23   1.69   2.62   3.52   5.23  2357 1.00
## z[7]      5.61    0.01 1.17   3.24   4.84   5.64   6.42   7.83  7229 1.00
## z[8]      6.96    0.01 1.13   4.66   6.22   6.99   7.70   9.17 18312 1.00
## z[9]      7.62    0.03 1.25   5.30   6.73   7.63   8.47  10.10  2246 1.00
## z[10]    11.13    0.03 1.37   8.41  10.19  11.15  12.11  13.63  1957 1.00
## z[11]    11.92    0.01 1.20   9.48  11.15  11.98  12.71  14.24  9023 1.00
## z[12]    12.61    0.01 1.21  10.15  11.81  12.65  13.45  14.89  7785 1.00
## z[13]    12.34    0.01 1.15  10.10  11.60  12.30  13.09  14.65 22148 1.00
## z[14]    12.75    0.01 1.14  10.46  12.02  12.79  13.50  14.97 20046 1.00
## z[15]    12.26    0.02 1.21   9.99  11.41  12.22  13.05  14.72  5923 1.00
## z[16]   13.14    0.01  1.15   10.83   12.41   13.16  13.88   15.42 15393 1.00
## z[17]   13.08    0.03  1.43   10.47   12.06   13.07  14.08   15.90  1703 1.00
## z[18]   16.05    0.01  1.17   13.67   15.28   16.08  16.86   18.26  7300 1.00
## z[19]   17.01    0.02  1.19   14.76   16.19   17.00  17.80   19.37  5060 1.00
## z[20]   19.82    0.01  1.17   17.44   19.03   19.86  20.60   22.07  8346 1.00
## z[21]   21.66    0.02  1.23   19.20   20.83   21.69  22.53   24.00  4286 1.00
## z[22]   22.18    0.01  1.13   19.93   21.47   22.17  22.90   24.44 21378 1.00
## z[23]   23.03    0.01  1.12   20.81   22.31   23.02  23.76   25.26 24790 1.00
## z[24]   24.25    0.01  1.15   21.97   23.49   24.29  25.05   26.45  9808 1.00
## z[25]   23.87    0.04  1.42   21.22   22.87   23.89  24.85   26.63  1517 1.00
## z[26]   26.83    0.01  1.15   24.55   26.09   26.87  27.58   29.08 19557 1.00
## z[27]   29.07    0.03  1.28   26.53   28.20   29.09  30.00   31.43  2417 1.00
## z[28]   29.12    0.01  1.13   26.95   28.36   29.09  29.88   31.41 15599 1.00
## z[29]   30.45    0.01  1.15   28.13   29.70   30.49  31.22   32.69 16379 1.00
## z[30]   30.66    0.01  1.21   28.31   29.86   30.60  31.44   33.11  8774 1.00
## z[31]   31.22    0.03  1.49   28.62   30.13   31.15  32.21   34.26  2205 1.00
## z[32]   33.57    0.04  1.48   30.95   32.50   33.53  34.59   36.55  1786 1.00
## z[33]   38.48    0.02  1.19   36.09   37.69   38.50  39.31   40.71  3714 1.00
## z[34]   41.32    0.01  1.15   39.03   40.58   41.30  42.07   43.58 18615 1.00
## z[35]   44.77    0.02  1.32   42.02   43.92   44.84  45.70   47.18  4716 1.00
## z[36]   47.14    0.02  1.39   44.20   46.21   47.22  48.14   49.67  3882 1.00
## z[37]   48.46    0.03  1.40   45.59   47.53   48.51  49.49   50.95  2921 1.00
## z[38]   47.90    0.01  1.13   45.69   47.17   47.87  48.63   50.19 21686 1.00
## z[39]   48.03    0.01  1.16   45.78   47.27   48.01  48.79   50.38 17313 1.00
## z[40]   48.25    0.03  1.35   45.79   47.28   48.23  49.18   50.97  2166 1.00
## z[41]   50.93    0.01  1.14   48.64   50.18   50.96  51.70   53.10 11438 1.00
## z[42]   52.11    0.01  1.13   49.93   51.37   52.07  52.85   54.38 18390 1.00
## z[43]   53.85    0.01  1.13   51.65   53.12   53.82  54.59   56.09 20960 1.00
## z[44]   56.00    0.01  1.13   53.74   55.28   56.01  56.72   58.23 23165 1.00
## z[45]   58.00    0.01  1.19   55.60   57.22   58.05  58.79   60.29 11970 1.00
## z[46]   59.41    0.02  1.20   56.96   58.61   59.45  60.26   61.67  5575 1.00
## z[47]   59.38    0.01  1.19   57.10   58.57   59.35  60.17   61.82 10145 1.00
## z[48]   60.47    0.02  1.25   58.16   59.56   60.44  61.31   63.02  4695 1.00
## z[49]   63.07    0.01  1.17   60.70   62.34   63.08  63.81   65.37 21511 1.00
## z[50]   65.46    0.02  1.40   62.58   64.54   65.52  66.41   68.08  7767 1.00
## v        1.36    0.00  0.28    0.79    1.18    1.35   1.53    1.91  8759 1.00
## lp__  -100.73    0.79 12.93 -119.54 -107.82 -102.11 -96.25 -66.96   271 1.01
##
## Samples were drawn using NUTS(diag_e) at Mon Jun  9 22:51:27 2025.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```
# Estimates
calc_rMSE(alpha = alpha, estimate = states)
```

```
## [1] 1.33184
```

```
# Observations
calc_rMSE(alpha = alpha, estimate = y)
```

```
## [1] 0.8895432
```

Here we seem to be doing worst. The Observations are somehow a better estimate than the actual state estimates we are using.

```r
# Number of states
N <- 50

# Nu
nu <- 1

# Process errors
omega <- rnorm(n = N, mean = 0, sd = 1)

# Initialize alpha vector
alpha <- rep(NA, N)
alpha[1] <- 1

# Store true latent states
for(i in 1:(length(alpha)-1)) {
    alpha[i+1] <- alpha[i] + nu + omega[i+1]
}

# Vector of observation errors
epsilon <- rnorm(n = N, mean = 0, sd = 3)

# Simulated observations
y <- alpha + epsilon
```

Again we will fit the model with stan and try to recover the parameters. The Basic_SSM2.stan file fits a linear SSM with observation error and process error.

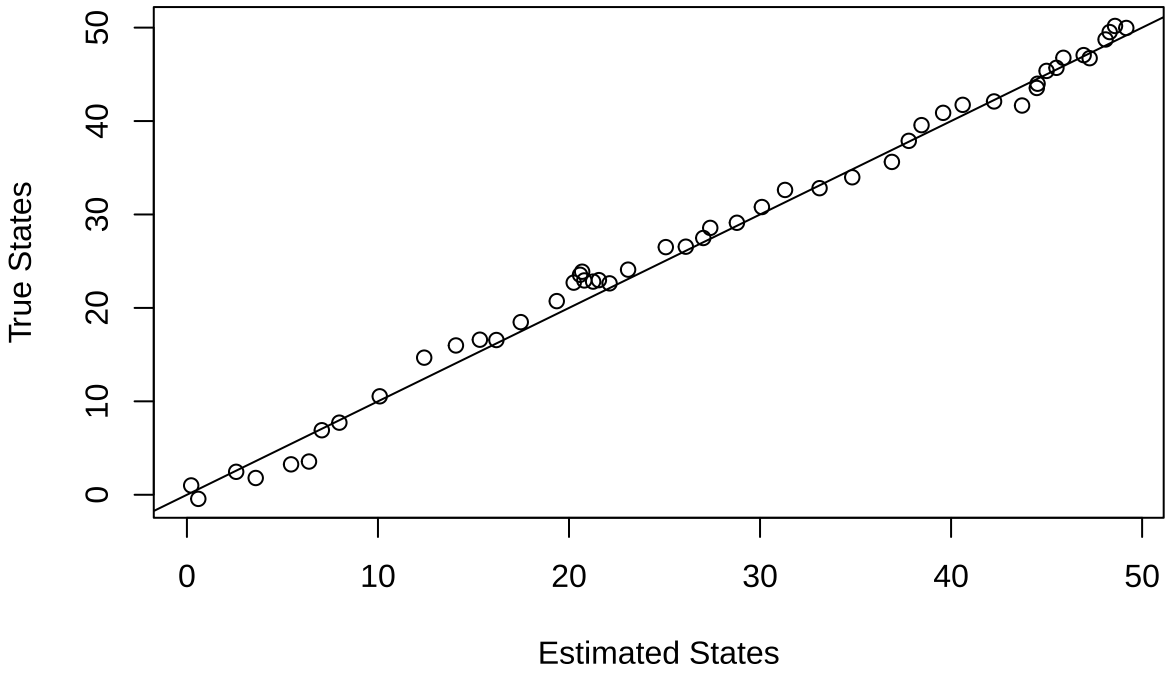
```r
data_stan <- list(TT = length(y), y = y, z0 = 0)

stan_test <- stan(file = "Basic_SSM2.stan",
                  data = data_stan,
                  chains = 3, iter = 5000)
```

```
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 5e-06 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.05 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 5000 [  0%]  (Warmup)
## Chain 1: Iteration:  500 / 5000 [ 10%]  (Warmup)
## Chain 1: Iteration: 1000 / 5000 [ 20%]  (Warmup)
## Chain 1: Iteration: 1500 / 5000 [ 30%]  (Warmup)
## Chain 1: Iteration: 2000 / 5000 [ 40%]  (Warmup)
## Chain 1: Iteration: 2500 / 5000 [ 50%]  (Warmup)
## Chain 1: Iteration: 2501 / 5000 [ 50%]  (Sampling)
## Chain 1: Iteration: 3000 / 5000 [ 60%]  (Sampling)
## Chain 1: Iteration: 3500 / 5000 [ 70%]  (Sampling)
## Chain 1: Iteration: 4000 / 5000 [ 80%]  (Sampling)
## Chain 1: Iteration: 4500 / 5000 [ 90%]  (Sampling)
## Chain 1: Iteration: 5000 / 5000 [100%]  (Sampling)
## Chain 1:
```

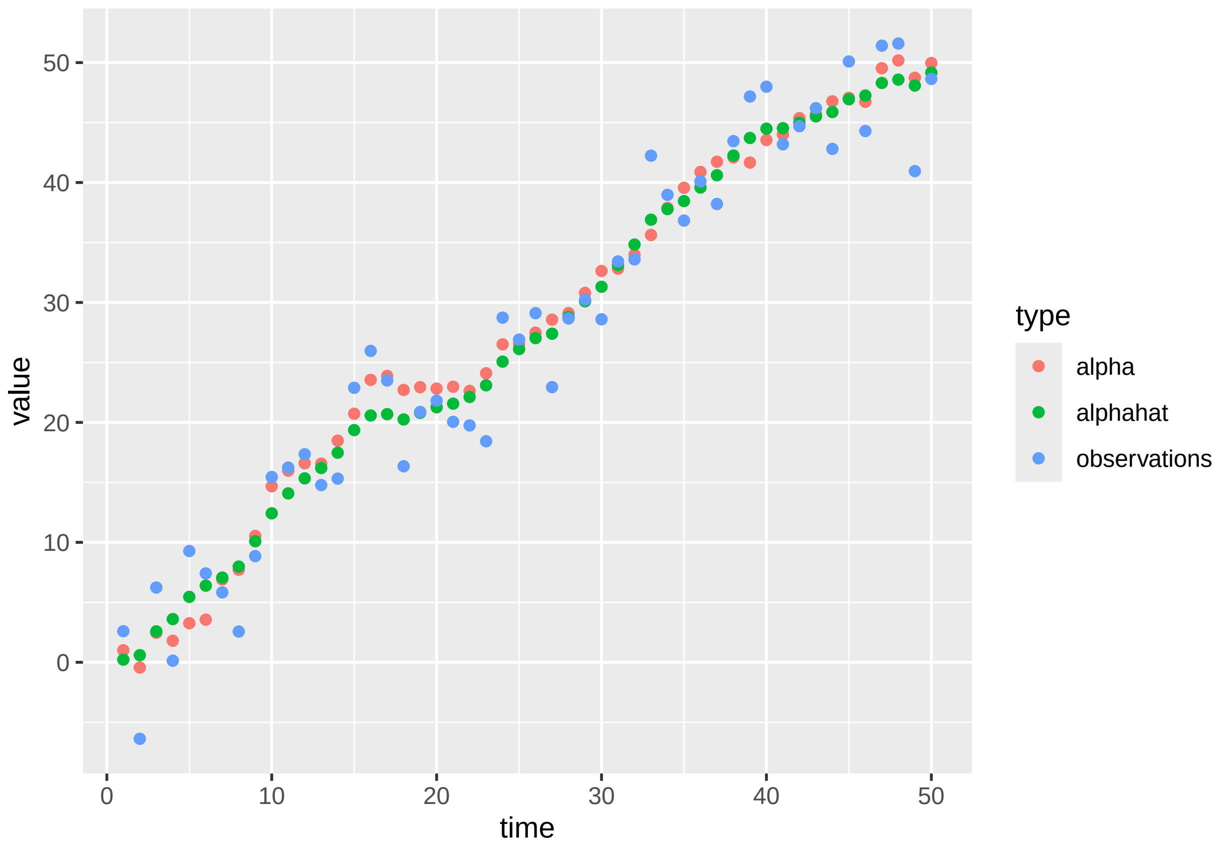
```
## Chain 1:  Elapsed Time: 0.339 seconds (Warm-up)
## Chain 1:                0.384 seconds (Sampling)
## Chain 1:                0.723 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 3e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.03 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 5000 [  0%]  (Warmup)
## Chain 2: Iteration:  500 / 5000 [ 10%]  (Warmup)
## Chain 2: Iteration: 1000 / 5000 [ 20%]  (Warmup)
## Chain 2: Iteration: 1500 / 5000 [ 30%]  (Warmup)
## Chain 2: Iteration: 2000 / 5000 [ 40%]  (Warmup)
## Chain 2: Iteration: 2500 / 5000 [ 50%]  (Warmup)
## Chain 2: Iteration: 2501 / 5000 [ 50%]  (Sampling)
## Chain 2: Iteration: 3000 / 5000 [ 60%]  (Sampling)
## Chain 2: Iteration: 3500 / 5000 [ 70%]  (Sampling)
## Chain 2: Iteration: 4000 / 5000 [ 80%]  (Sampling)
## Chain 2: Iteration: 4500 / 5000 [ 90%]  (Sampling)
## Chain 2: Iteration: 5000 / 5000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.431 seconds (Warm-up)
## Chain 2:                0.419 seconds (Sampling)
## Chain 2:                0.85 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 4e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.04 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 5000 [  0%]  (Warmup)
## Chain 3: Iteration:  500 / 5000 [ 10%]  (Warmup)
## Chain 3: Iteration: 1000 / 5000 [ 20%]  (Warmup)
## Chain 3: Iteration: 1500 / 5000 [ 30%]  (Warmup)
## Chain 3: Iteration: 2000 / 5000 [ 40%]  (Warmup)
## Chain 3: Iteration: 2500 / 5000 [ 50%]  (Warmup)
## Chain 3: Iteration: 2501 / 5000 [ 50%]  (Sampling)
## Chain 3: Iteration: 3000 / 5000 [ 60%]  (Sampling)
## Chain 3: Iteration: 3500 / 5000 [ 70%]  (Sampling)
## Chain 3: Iteration: 4000 / 5000 [ 80%]  (Sampling)
## Chain 3: Iteration: 4500 / 5000 [ 90%]  (Sampling)
## Chain 3: Iteration: 5000 / 5000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.364 seconds (Warm-up)
## Chain 3:                0.349 seconds (Sampling)
## Chain 3:                0.713 seconds (Total)
## Chain 3:
```

```
## Warning: There were 3 chains where the estimated Bayesian Fraction of Missing Information was low. Se
## https://mc-stan.org/misc/warnings.html#bfmi-low

## Warning: Examine the pairs() plot to diagnose sampling problems
```

```r
states <- colMeans(extract(stan_test, pars = c("z"))[[1]])

plot(x = states, y = alpha,
     xlab = "Estimated States",
     ylab = "True States")
abline(0,1)
```



```r
# Include a plot of time vs. estimated state, true state, and observations
time <- seq(from = 1, to = 50, by = 1)

df1 <- tibble(time = time,
              alpha = alpha,
              alphahat = states,
              observations = y)


df_long <- df1 %>%
  pivot_longer(cols = c(alpha, alphahat,observations), names_to = "type", values_to = "value")

ggplot(df_long, aes(x = time, y = value, color = type)) +
  geom_point()
```



```
stan_test
```

```
## Inference for Stan model: anon_model.
## 3 chains, each with iter=5000; warmup=2500; thin=1;
## post-warmup draws per chain=2500, total post-warmup draws=7500.
##
##          mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
## sdo      3.50    0.01 0.48   2.58   3.18   3.48   3.80   4.50  1613 1.00
## sdp      1.52    0.03 0.63   0.55   1.05   1.43   1.87   2.99   361 1.01
## z[1]     0.22    0.02 1.73  -3.23  -0.91   0.25   1.37   3.58  6765 1.00
## z[2]     0.60    0.04 1.84  -3.28  -0.57   0.66   1.85   4.05  1720 1.00
## z[3]     2.58    0.02 1.63  -0.78   1.51   2.62   3.65   5.73  6030 1.00
## z[4]     3.60    0.02 1.65   0.20   2.55   3.64   4.69   6.73  4621 1.00
## z[5]     5.45    0.02 1.62   2.25   4.37   5.46   6.51   8.69  5598 1.00
## z[6]     6.39    0.02 1.55   3.36   5.35   6.40   7.42   9.49  6170 1.00
## z[7]     7.06    0.03 1.64   3.73   6.00   7.11   8.16  10.12  2727 1.00
## z[8]     7.98    0.04 1.76   4.19   6.89   8.07   9.20  11.12  1551 1.00
## z[9]    10.09    0.02 1.61   6.92   9.08  10.10  11.14  13.17  5956 1.00
## z[10]   12.42    0.03 1.64   9.35  11.34  12.34  13.46  15.84  3316 1.00
## z[11]   14.08    0.04 1.70  10.94  12.91  14.01  15.16  17.64  2260 1.00
## z[12]   15.33    0.03 1.67  12.17  14.21  15.27  16.42  18.74  2468 1.00
## z[13]   16.20    0.03 1.62  13.12  15.11  16.15  17.23  19.60  3982 1.00
## z[14]   17.48    0.03 1.67  14.35  16.33  17.43  18.55  20.93  3473 1.00
## z[15]   19.36    0.06 1.89  15.97  18.02  19.25  20.57  23.29  1147 1.00
## z[16]   20.58    0.07 2.02  17.01  19.13  20.44  21.87  24.93   953 1.00
## z[17]   20.69    0.04 1.75  17.44  19.48  20.60  21.81  24.28  1722 1.00
## z[18]   20.25    0.02 1.59  16.91  19.24  20.27  21.27  23.40  5577 1.00
## z[19]   20.80    0.02  1.57   17.55   19.80   20.82   21.81  23.92  6196 1.00
## z[20]   21.26    0.02  1.59   18.02   20.22   21.32   22.31  24.34  4754 1.00
## z[21]   21.56    0.03  1.66   18.07   20.56   21.64   22.68  24.69  2297 1.00
## z[22]   22.12    0.04  1.76   18.47   21.00   22.22   23.32  25.33  1586 1.00
## z[23]   23.09    0.04  1.75   19.50   22.00   23.19   24.29  26.29  1723 1.00
## z[24]   25.07    0.02  1.61   21.89   24.02   25.07   26.11  28.25  6887 1.00
## z[25]   26.11    0.02  1.57   22.95   25.07   26.14   27.14  29.20  6950 1.00
## z[26]   27.03    0.02  1.59   23.89   26.00   27.06   28.06  30.12  6438 1.00
## z[27]   27.39    0.04  1.68   23.86   26.33   27.50   28.55  30.47  1826 1.00
## z[28]   28.79    0.03  1.59   25.53   27.76   28.83   29.85  31.81  3349 1.00
## z[29]   30.10    0.02  1.59   26.89   29.04   30.17   31.16  33.11  4057 1.00
## z[30]   31.31    0.03  1.62   27.91   30.30   31.38   32.37  34.41  3167 1.00
## z[31]   33.12    0.02  1.54   30.06   32.11   33.11   34.13  36.19  6225 1.00
## z[32]   34.83    0.02  1.56   31.82   33.82   34.79   35.83  38.02  6222 1.00
## z[33]   36.90    0.05  1.73   33.76   35.71   36.81   37.94  40.63  1426 1.00
## z[34]   37.79    0.03  1.61   34.81   36.67   37.71   38.81  41.16  2141 1.00
## z[35]   38.45    0.02  1.56   35.46   37.41   38.41   39.49  41.57  6085 1.00
## z[36]   39.58    0.02  1.58   36.53   38.53   39.53   40.61  42.74  5696 1.00
## z[37]   40.61    0.02  1.60   37.46   39.55   40.61   41.66  43.79  5228 1.00
## z[38]   42.25    0.03  1.66   39.22   41.13   42.18   43.32  45.70  2339 1.00
## z[39]   43.72    0.05  1.82   40.43   42.44   43.61   44.88  47.57  1254 1.00
## z[40]   44.49    0.05  1.77   41.24   43.25   44.40   45.65  48.14  1433 1.00
## z[41]   44.53    0.02  1.60   41.45   43.44   44.47   45.57  47.76  4422 1.00
## z[42]   45.00    0.02  1.58   41.87   43.97   44.98   46.02  48.15  5569 1.00
## z[43]   45.51    0.02  1.56   42.40   44.50   45.52   46.56  48.58  5927 1.00
## z[44]   45.88    0.02  1.60   42.63   44.87   45.94   46.90  48.93  5137 1.00
## z[45]   46.94    0.02  1.62   43.74   45.90   46.92   47.98  50.20  5388 1.00
## z[46]   47.25    0.02  1.64   43.96   46.19   47.27   48.31  50.52  5278 1.00
## z[47]   48.30    0.02  1.70   44.97   47.20   48.30   49.39  51.66  5119 1.00
## z[48]   48.58    0.03  1.74   45.10   47.44   48.61   49.74  52.01  4239 1.00
## z[49]   48.09    0.06  2.14   43.58   46.70   48.17   49.57  51.98  1131 1.00
## z[50]   49.17    0.05  2.26   44.60   47.68   49.22   50.75  53.38  1761 1.00
## v        1.00    0.00  0.25    0.47    0.87    1.01    1.14   1.50  8610 1.00
## lp__  -126.89    0.94 17.19 -155.61 -139.33 -128.33 -115.87 -88.41   335 1.01
##
## Samples were drawn using NUTS(diag_e) at Mon Jun  9 22:51:31 2025.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```r
# Estimates
calc_rMSE(alpha = alpha, estimate = states)
```

```
## [1] 1.350824
```

```r
# Observations
calc_rMSE(alpha = alpha, estimate = y)
```

```
## [1] 3.256957
```

Here we are doing decently again.

Conclusion

It seems like we do a halfway decent job of getting back to the true states when the process and observation errors are similar or when the process error is small and the observation error is large. We do worst when the process error is large but the observation error is small.

If this is an avenue worth exploring more I could write some code to repeatedly fit these models on different samples and determine what the rMSE typically is for both the states and the slope/error parameters.

Observation Error and Process Error for both the states and the slope

The Last model I'd like to try is as follows:

$$
\begin{aligned}
y_t &= \alpha_t + \epsilon_t && (7)\\
\alpha_{t+1} &= \alpha_t + \nu_t + \omega_t && (8)\\
\nu_{t+1} &= \nu_t + \zeta_t && (9)
\end{aligned}
$$

where $\epsilon_t \sim N(0,5)$, $\omega_t \sim N(0,2)$ and $\zeta_t \sim N(0,1)$. We'll start with $\alpha_1 = 1$ and $\nu_1 = 1$

```r
# Number of states
N <- 50

# Initialize Nu vector
nu <- rep(NA,N)
nu[1] <- 1

# Initialize alpha vector
alpha <- rep(NA, N)
alpha[1] <- 1

# Process errors
omega <- rnorm(n = N, mean = 0, sd = 2)
zeta <- rnorm(n = N, mean = 0, sd = .5)



# Store true latent states
for(i in 1:(length(alpha)-1)) {
    nu[i+1] <- nu[i] + zeta[i]
    alpha[i+1] <- alpha[i] + nu[i] + omega[i+1]
}

# Vector of observation errors
epsilon <- rnorm(n = N, mean = 0, sd = 5)

# Simulated observations
y <- alpha + epsilon
```

Again we will fit the model with stan and try to recover the parameters. The Basic_SSM3.stan file fits a linear SSM with observation error and process error and a slope that follows a random walk. I included some informative priors in the stan code but it still seems to do okay even without them.

```r
data_stan <- list(TT = length(y), y = y, z0 = 1)

stan_test <- stan(file = "Basic_SSM3.stan",
                  data = data_stan,
                  chains = 3, iter = 10000)
```

```
## Trying to compile a simple C file

## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## using C compiler: ‘Apple clang version 17.0.0 (clang-1700.0.13.3)’
## using SDK: ‘MacOSX15.4.sdk’
## clang -arch arm64 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I"/Library/Frame
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/StanHeade
## In file included from /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/RcppEigen
## In file included from /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/RcppEigen
## /Library/Frameworks/R.framework/Versions/4.4-arm64/Resources/library/RcppEigen/include/Eigen/src/Cor
##   679 | #include <cmath>
##       |          ^~~~~~~
## 1 error generated.
## make: *** [foo.o] Error 1
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 4.3e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.43 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 1: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 1: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 1: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 1: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 1: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 1: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 1: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 1: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 1: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 1: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 1: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 1.64 seconds (Warm-up)
## Chain 1:                1.56 seconds (Sampling)
## Chain 1:                3.2 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 5e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.05 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 2: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 2: Iteration: 2000 / 10000 [ 20%]  (Warmup)
```

```
## Chain 2: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 2: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 2: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 2: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 2: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 2: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 2: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 2: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 2: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 2.631 seconds (Warm-up)
## Chain 2:                1.96 seconds (Sampling)
## Chain 2:                4.591 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 5e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.05 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 3: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 3: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 3: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 3: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 3: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 3: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 3: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 3: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 3: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 3: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 3: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 1.616 seconds (Warm-up)
## Chain 3:                2.039 seconds (Sampling)
## Chain 3:                3.655 seconds (Total)
## Chain 3:
```

```
## Warning: There were 3 divergent transitions after warmup. See
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.
```

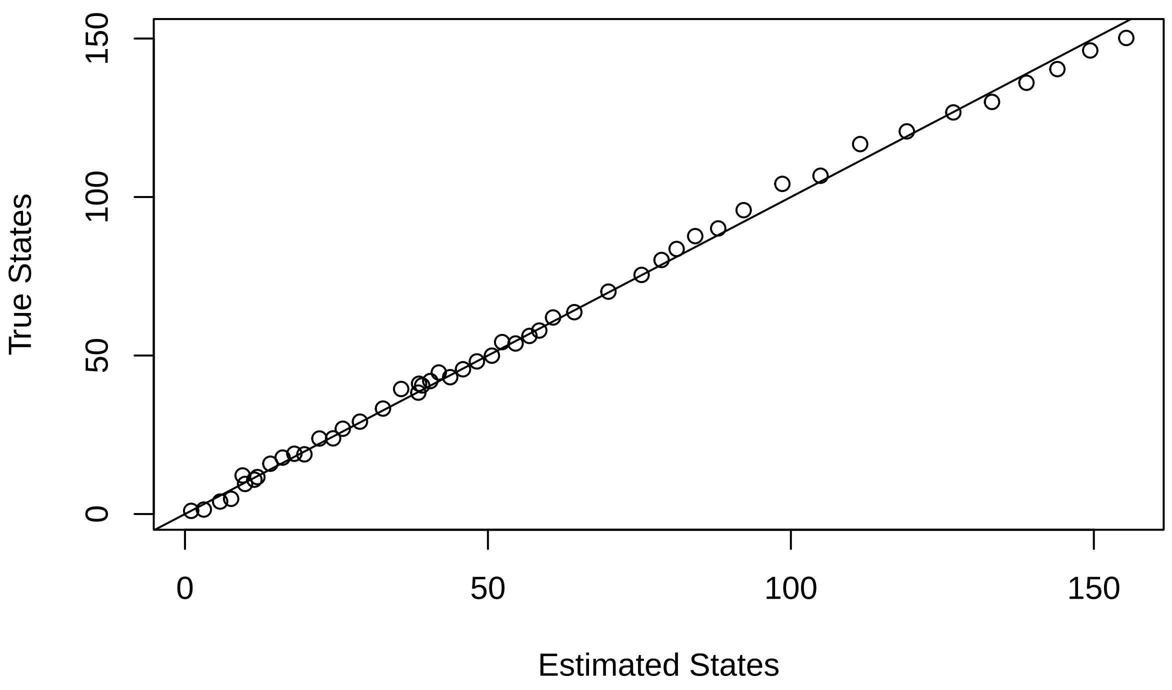
```
## Warning: There were 3 chains where the estimated Bayesian Fraction of Missing Information was low. S
## https://mc-stan.org/misc/warnings.html#bfmi-low
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

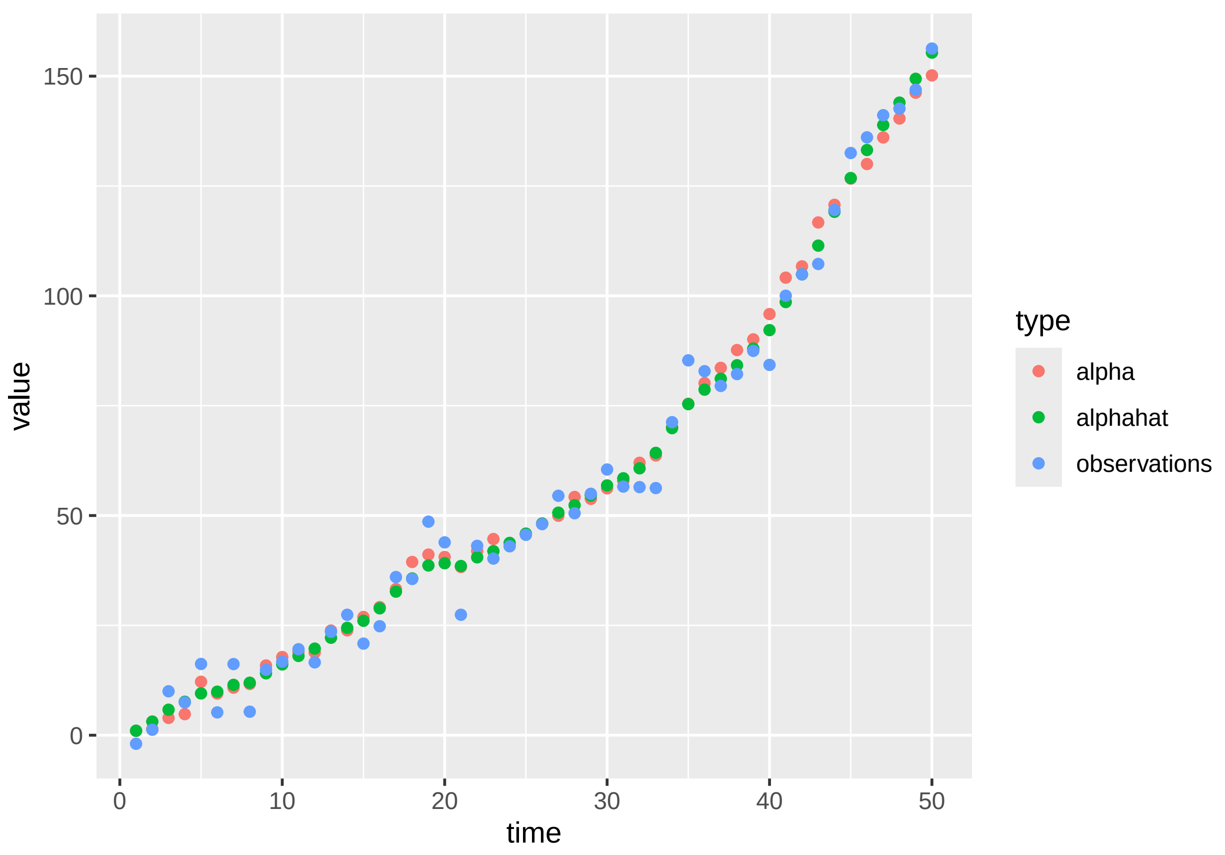
```
## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quant
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#tail-ess
```

```r
states <- colMeans(extract(stan_test, pars = c("z"))[[1]])
nuhat <- colMeans(extract(stan_test, pars = c("v"))[[1]])
```

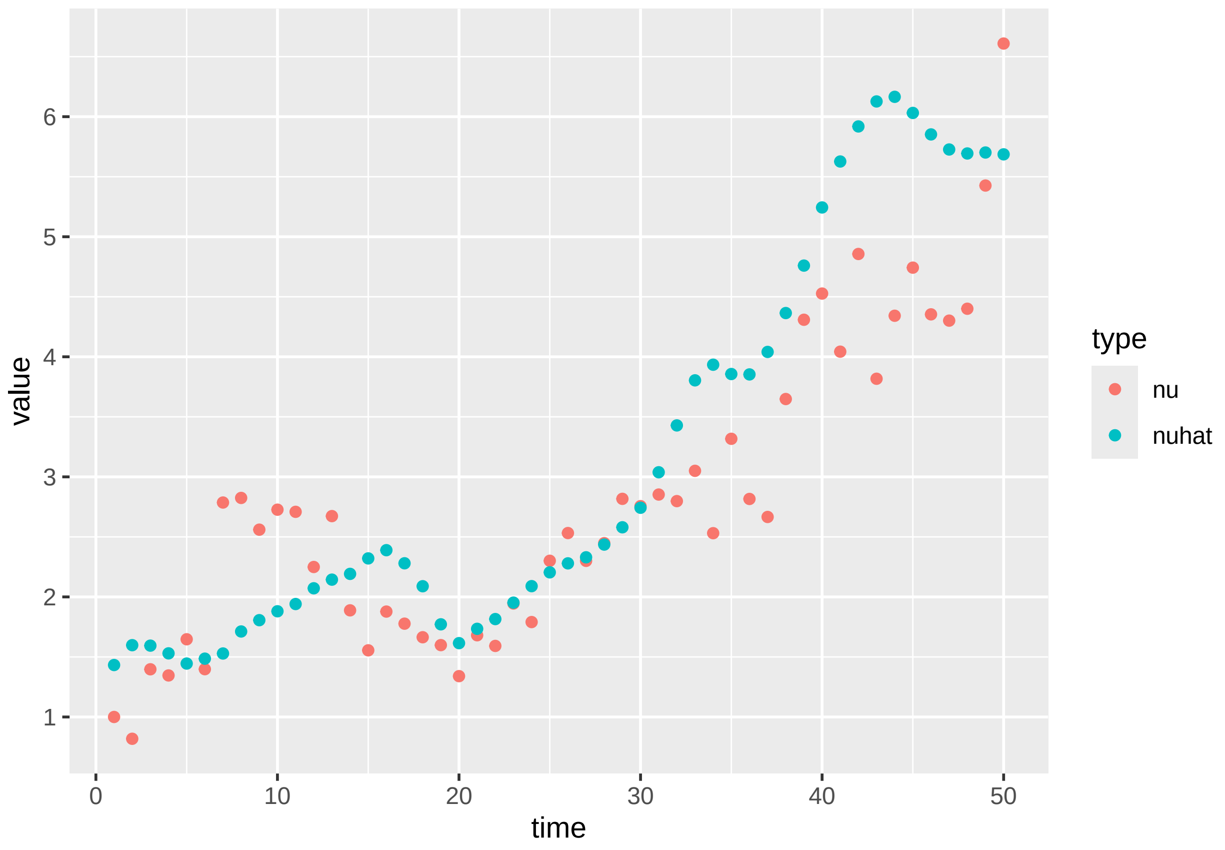
```r
plot(x = states, y = alpha,
     xlab = "Estimated States",
     ylab = "True States")
abline(0,1)
```



```r
# Include a plot of time vs. estimated state, true state, and observations
time <- seq(from = 1, to = 50, by = 1)

df1 <- tibble(time = time,
              alpha = alpha,
              alphahat = states,
              observations = y,
              nu = nu,
              nuhat = nuhat)


df_long <- df1 %>%
  pivot_longer(cols = c(alpha, alphahat, observations), names_to = "type", values_to = "value")

df_long2 <- df1 %>%
  pivot_longer(cols = c(nu, nuhat), names_to = "type", values_to = "value")

ggplot(df_long, aes(x = time, y = value, color = type)) +
  geom_point()
```



```r
ggplot(df_long2, aes(x = time, y = value, color = type)) +
  geom_point()
```



```
stan_test
```

```
## Inference for Stan model: anon_model.
## 3 chains, each with iter=10000; warmup=5000; thin=1;
## post-warmup draws per chain=5000, total post-warmup draws=15000.
##
##          mean se_mean   sd  2.5%   25%   50%   75% 97.5% n_eff Rhat
## sdo      4.96    0.01 0.58  3.87  4.57  4.95  5.34  6.13  3629 1.00
## sdp      2.14    0.04 0.85  0.46  1.55  2.16  2.71  3.79   446 1.01
## sdv      0.74    0.01 0.35  0.28  0.49  0.66  0.90  1.63   559 1.00
## z[1]     1.02    0.01 0.94 -0.82  0.38  1.02  1.66  2.87 14739 1.00
## z[2]     3.11    0.01 1.88 -0.61  1.90  3.07  4.30  6.90 17231 1.00
## z[3]     5.80    0.03 2.26  1.63  4.26  5.71  7.24 10.51  5843 1.00
## z[4]     7.62    0.03 2.37  3.15  6.02  7.53  9.18 12.46  5964 1.00
## z[5]     9.51    0.04 2.48  4.93  7.82  9.43 11.13 14.65  4206 1.00
## z[6]     9.91    0.03 2.42  5.18  8.30  9.92 11.50 14.65  8503 1.00
## z[7]    11.46    0.04 2.46  6.71  9.81 11.48 13.08 16.32  4376 1.00
## z[8]    11.95    0.03 2.52  6.85 10.29 11.99 13.60 16.81  6399 1.00
## z[9]    14.09    0.02 2.45  9.29 12.47 14.09 15.71 18.91 11246 1.00
## z[10]   16.11    0.02 2.42 11.30 14.51 16.12 17.71 20.88 11657 1.00
## z[11]   18.05    0.03 2.43 13.19 16.44 18.03 19.68 22.77  7670 1.00
## z[12]   19.71    0.03 2.45 14.90 18.09 19.73 21.29 24.56  8286 1.00
## z[13]   22.19    0.03 2.44 17.50 20.58 22.15 23.79 26.97  6144 1.00
## z[14]   24.45    0.02 2.42 19.72 22.82 24.45 26.05 29.21  9697 1.00
## z[15]   26.05    0.03 2.47 21.07 24.48 26.08 27.71 30.83  7784 1.00
## z[16]   28.88    0.03 2.45 23.99 27.25 28.91 30.52 33.65  8659 1.00
## z[17]   32.68    0.04 2.45 28.01 31.00 32.66 34.31 37.62  4845 1.00
## z[18]   35.68    0.05  2.55   30.84   33.91   35.63   37.36   40.82  3186 1.00
## z[19]   38.64    0.06  2.83   33.46   36.68   38.49   40.49   44.43  2077 1.00
## z[20]   39.14    0.03  2.54   34.23   37.42   39.10   40.80   44.17  5292 1.00
## z[21]   38.51    0.03  2.53   33.31   36.88   38.57   40.21   43.40  7680 1.00
## z[22]   40.49    0.02  2.44   35.64   38.91   40.49   42.11   45.29 11239 1.00
## z[23]   41.89    0.02  2.48   36.93   40.26   41.95   43.56   46.65 11346 1.00
## z[24]   43.77    0.02  2.44   38.87   42.21   43.79   45.37   48.51 12424 1.00
## z[25]   45.89    0.02  2.44   41.02   44.29   45.91   47.51   50.71 12535 1.00
## z[26]   48.18    0.02  2.45   43.42   46.60   48.15   49.78   53.03 11998 1.00
## z[27]   50.65    0.03  2.49   45.81   49.03   50.64   52.34   55.54  9395 1.00
## z[28]   52.34    0.03  2.47   47.58   50.65   52.34   53.99   57.18  7259 1.00
## z[29]   54.56    0.03  2.44   49.76   52.95   54.53   56.19   59.34  7966 1.00
## z[30]   56.85    0.03  2.40   52.15   55.25   56.84   58.47   61.53  8769 1.00
## z[31]   58.48    0.03  2.45   53.58   56.88   58.52   60.13   63.22  7135 1.00
## z[32]   60.75    0.04  2.59   55.42   59.06   60.84   62.52   65.56  4197 1.00
## z[33]   64.26    0.04  2.57   59.03   62.60   64.35   66.00   68.97  4385 1.00
## z[34]   69.89    0.02  2.44   65.05   68.31   69.88   71.45   74.73 11630 1.00
## z[35]   75.36    0.05  2.78   70.08   73.50   75.22   77.13   81.09  2786 1.00
## z[36]   78.65    0.04  2.62   73.63   76.91   78.59   80.37   84.07  3840 1.00
## z[37]   81.14    0.02  2.47   76.32   79.51   81.11   82.79   86.01 13498 1.00
## z[38]   84.21    0.03  2.50   79.22   82.57   84.26   85.92   88.92 10019 1.00
## z[39]   88.00    0.04  2.55   82.85   86.30   88.03   89.72   92.91  5138 1.00
## z[40]   92.20    0.05  2.71   86.66   90.42   92.24   94.04   97.39  3357 1.00
## z[41]   98.59    0.04  2.52   93.51   96.89   98.62  100.30  103.41  4854 1.00
## z[42]  104.89    0.03  2.45  100.02  103.28  104.87  106.55  109.67  8314 1.00
## z[43]  111.44    0.02  2.46  106.43  109.84  111.45  113.09  116.15 11279 1.00
## z[44]  119.13    0.03  2.42  114.37  117.49  119.14  120.75  123.93  8922 1.00
## z[45]  126.81    0.05  2.62  121.80  125.02  126.75  128.56  132.17  3172 1.00
## z[46]  133.19    0.05  2.65  128.16  131.40  133.12  134.94  138.42  2651 1.00
## z[47]  138.88    0.05  2.59  133.84  137.17  138.82  140.56  144.24  2563 1.00
## z[48]  143.98    0.03  2.59  138.90  142.28  143.99  145.71  148.99  5790 1.00
## z[49]  149.40    0.03  2.84  143.75  147.50  149.44  151.33  155.01 10678 1.00
## z[50]  155.35    0.05  3.59  148.39  153.03  155.35  157.75  162.45  5283 1.00
## v[1]     1.43    0.01  0.77   -0.11    0.93    1.45    1.95    2.93 11018 1.00
## v[2]     1.60    0.01  0.90   -0.20    1.04    1.60    2.16    3.40  8702 1.00
## v[3]     1.59    0.01  0.92   -0.24    1.03    1.59    2.15    3.43  8193 1.00
## v[4]     1.53    0.01  0.94   -0.45    0.96    1.54    2.10    3.40  6547 1.00
## v[5]     1.44    0.02  0.96   -0.56    0.87    1.47    2.06    3.29  3598 1.00
## v[6]     1.49    0.01  0.95   -0.46    0.92    1.51    2.08    3.32  4975 1.00
## v[7]     1.53    0.01  0.95   -0.44    0.95    1.55    2.13    3.38  7508 1.00
## v[8]     1.71    0.01  0.96   -0.23    1.12    1.72    2.30    3.67  5983 1.00
## v[9]     1.81    0.01  0.96   -0.12    1.23    1.81    2.38    3.75 11350 1.00
## v[10]    1.88    0.01  0.95   -0.07    1.31    1.88    2.45    3.81 11260 1.00
## v[11]    1.94    0.01  0.95    0.02    1.36    1.93    2.51    3.88 10155 1.00
## v[12]    2.07    0.01  0.96    0.16    1.50    2.05    2.64    4.06 10316 1.00
## v[13]    2.14    0.01  0.98    0.24    1.53    2.11    2.71    4.28  5380 1.00
## v[14]    2.19    0.01  0.97    0.32    1.60    2.17    2.76    4.24  5866 1.00
## v[15]    2.32    0.01  0.98    0.45    1.71    2.28    2.89    4.44  6882 1.00
## v[16]    2.39    0.02  1.01    0.53    1.73    2.32    2.97    4.63  3244 1.00
## v[17]    2.28    0.01  0.97    0.41    1.67    2.25    2.86    4.33  6766 1.00
## v[18]    2.09    0.01  0.95    0.18    1.50    2.08    2.67    4.01  9218 1.00
## v[19]    1.77    0.01  0.99   -0.38    1.18    1.82    2.40    3.64  5331 1.00
## v[20]    1.61    0.02  1.04   -0.68    1.01    1.69    2.29    3.49  2827 1.00
## v[21]    1.73    0.01  0.99   -0.36    1.16    1.78    2.38    3.58  4695 1.00
## v[22]    1.82    0.01  0.99   -0.26    1.22    1.86    2.44    3.70  5065 1.00
## v[23]    1.95    0.02  0.99   -0.11    1.35    1.98    2.59    3.81  4229 1.00
## v[24]    2.09    0.01  0.97    0.08    1.50    2.11    2.71    3.99  7948 1.00
## v[25]    2.20    0.01  0.97    0.21    1.60    2.22    2.82    4.11  6759 1.00
## v[26]    2.28    0.01  0.96    0.32    1.69    2.29    2.89    4.12  6610 1.00
## v[27]    2.33    0.02  0.97    0.31    1.74    2.36    2.96    4.22  3953 1.00
## v[28]    2.44    0.01  0.98    0.42    1.85    2.46    3.05    4.35  4746 1.00
## v[29]    2.58    0.02  0.99    0.45    1.98    2.62    3.22    4.45  3634 1.00
## v[30]    2.74    0.01  0.98    0.69    2.16    2.77    3.38    4.57  5507 1.00
## v[31]    3.04    0.01  0.96    1.05    2.45    3.06    3.64    4.88  5572 1.00
## v[32]    3.43    0.01  0.95    1.56    2.83    3.41    4.01    5.38  9005 1.00
## v[33]    3.80    0.01  0.98    1.96    3.18    3.75    4.38    5.90  5482 1.00
## v[34]    3.93    0.01  0.98    2.11    3.32    3.89    4.50    6.03  6065 1.00
## v[35]    3.86    0.01  0.95    1.93    3.28    3.86    4.45    5.74 10604 1.00
## v[36]    3.85    0.02  0.98    1.77    3.26    3.91    4.48    5.68  3538 1.00
## v[37]    4.04    0.02  1.01    1.87    3.45    4.09    4.68    5.91  4039 1.00
## v[38]    4.36    0.01  1.00    2.29    3.77    4.40    4.98    6.29  6580 1.00
## v[39]    4.76    0.01  1.00    2.72    4.17    4.77    5.37    6.74  8059 1.00
## v[40]    5.24    0.01  1.03    3.27    4.59    5.23    5.86    7.39  5295 1.00
## v[41]    5.63    0.02  1.08    3.56    4.92    5.58    6.30    7.85  2013 1.00
## v[42]    5.92    0.03  1.14    3.76    5.16    5.86    6.60    8.40  1208 1.00
## v[43]    6.13    0.03  1.18    3.91    5.35    6.08    6.84    8.61  1188 1.00
## v[44]    6.17    0.04  1.18    3.92    5.38    6.12    6.87    8.70  1116 1.00
## v[45]    6.03    0.03  1.12    3.87    5.29    6.02    6.75    8.27  1450 1.00
## v[46]    5.85    0.02  1.11    3.62    5.15    5.86    6.57    8.04  3531 1.00
## v[47]    5.73    0.02  1.20    3.24    4.97    5.74    6.50    8.06  3279 1.00
## v[48]    5.69    0.03  1.34    2.92    4.85    5.71    6.56    8.33  2078 1.00
## v[49]    5.70    0.05  1.58    2.61    4.76    5.72    6.68    8.76  1063 1.00
## v[50]    5.69    0.06  1.78    1.92    4.65    5.72    6.79    9.09   977 1.00
## lp__  -164.60    1.50 28.72 -218.11 -184.32 -165.83 -146.39 -102.80   364 1.01
##
## Samples were drawn using NUTS(diag_e) at Mon Jun  9 22:51:57 2025.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

```r
# Estimates
calc_rMSE(alpha = alpha, estimate = states)
```

```
## [1] 2.211662
```

```r
# Observations
calc_rMSE(alpha = alpha, estimate = y)
```

```
## [1] 4.788107
```