Meeting 25

Hi Martin,  
  
While waiting for the train I was just thinking about the second setting we considered for analysis. It seems that, in that case, the role of $$a$$ is irrelevant (as we can think of our observations effectively as Y\_t-X\_{1,t}). So, it is really $b$ that is playing the crucial role (in the parameterization we had).  
  
All the best,  
  
Rui

Hi Martin,

Next to that, something about the cross validation approach: My feeling is still that cross validation works for us because the noise is independent. Is there any sensible way of trying to confirm this? One way would be to increase/decrease the noise level. Could even try to do it extreme and go to a deterministic process, I am really curious  to see if cross validation still makes sense then. Alternatively we could try to see what happens with dependent noise (where the noise is a random walk by itself). I am not sure though if the resulting process is sensible.

Best,

Alex

# AR(2) With a = 0 setting.

Now four different model types:

1. AR(0), so X\_t = epsilon\_t,
2. AR(1), so X\_t = aX\_t-1 + epsilon\_t,
3. AR(2), so X\_t = aX\_t-1 + bX\_t-2 + epsilon\_t,
4. AR(2 – 1), so X\_t = bX\_t-1 + epsilon\_t.

True a: 0. True b: 0. Time T: 1001.

True Risk or CV AR(0): 0.9904.

LS Estimates a, b: [-0.028 0.012].

Corresponding emprirical risk (MSE^2): 0.9883.

CV AR(2): 0.994.

LS estimate a: -0.03.

Corresponding emprirical risk (MSE^2): 0.9886.

CV AR(1): 0.9915.

LS estimate b: 0.01.

Corresponding emprirical risk (MSE^2): 0.9907.

CV AR(2 - 1): 0.9928.

In this case, the correct one is picked, followed by AR(1), AR(2 – 1), AR(2), as expected. However, the correct model is not always chosen! Let us consider 1000 simulations, and see the proportion of models picked.

T = 11: [567, 147, 69, 217] / [2988, 697, 351, 964]

T = 26: [618, 136, 60, 186] / [3116, 650, 255, 979]

T = 51: [635, 127, 52, 186] / [3170, 637, 216, 977]

T = 76: [666, 128, 33, 173] / [3175, 660, 197, 968]

T = 101: [640, 133, 39, 188] /

# Mixture of ARs with fixed 1.

Now, we fix a to be equal to 1. Naturally, although a = 1 is indeed the data generating value, the empirical risk where we can estimate a will be smaller.

True a: 0.8. True b: 0.9. Time T: 100.

LS Estimates a, b: [0.96 0.03].

Emprirical risk: 1.112.

CV using X1 and X2: 1.1485.

LS estimate a: 0.97.

Emprirical risk: 1.1164.

CV Using X1: 1.1358.

LS estimate b: 0.17.

Emprirical risk: 4.8347.

CV Using X2: 4.9233.

LS estimate b, a = 1: 0.03.

Emprirical risk: 1.1174.

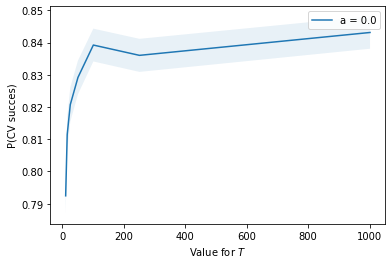
CV Using X2, a = 1: 1.1355.

CV Using X1, a = 1: **1.1205.**

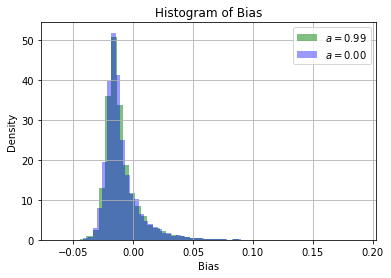
Do we always find this one to be the smallest one?

## Dependence on a.

Does not matter at all, as Rui discussed as well. Analytically easily defendable.



[0.79245, 0.81145, 0.82075, 0.8292, 0.83925, 0.83605, 0.84315], 20,000.



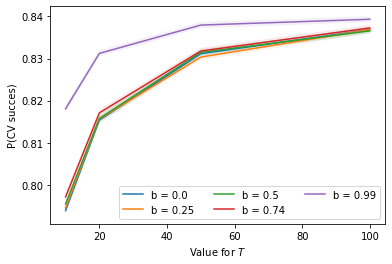
## Dependence on T.

Dependence on T seem to be as follows. The probability of CV picking the right model seems to increase when T increases. Especially for the [0, 100] region, the success increases from ~0.795 to ~0.835. From thereon out, T is not very influential.

## 

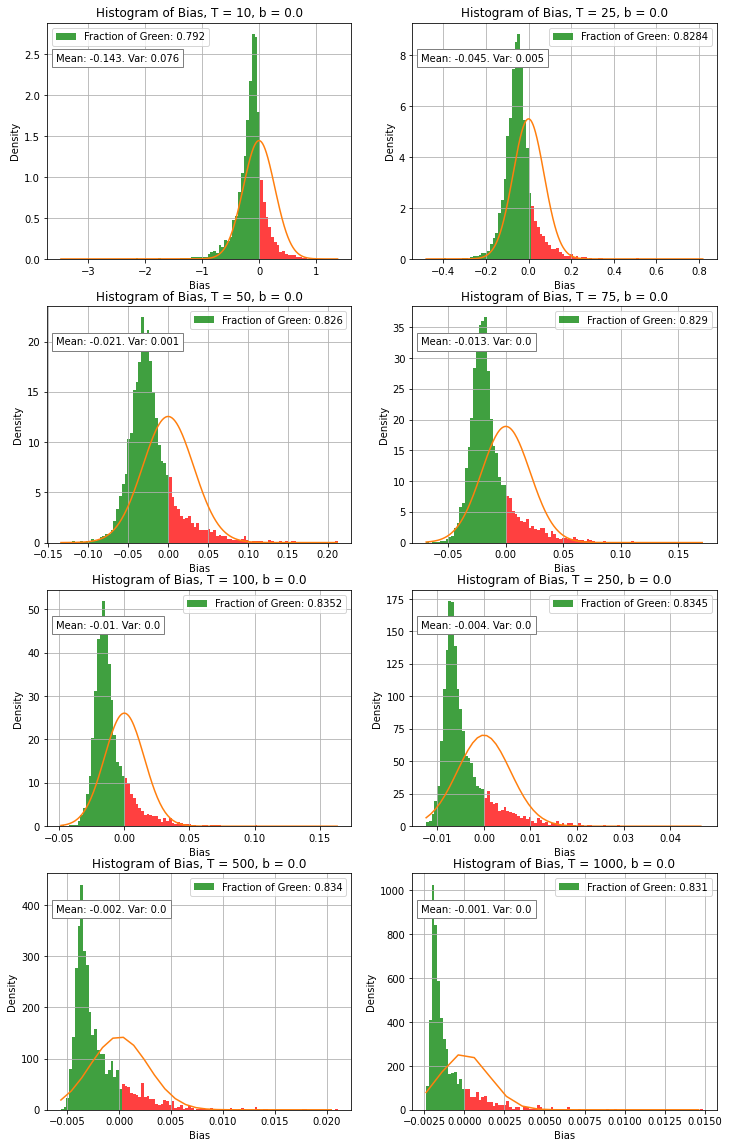
We see a clear difference for different values for *T*. The larger *T* is, the larger the probability of success. Interestingly, the value for *b* does also not matter then. We also here see the sharp increase for when *b* goes from 0.75 to 0.99. This increase is much less sharp for larger *T*.

## Dependence on *b*.

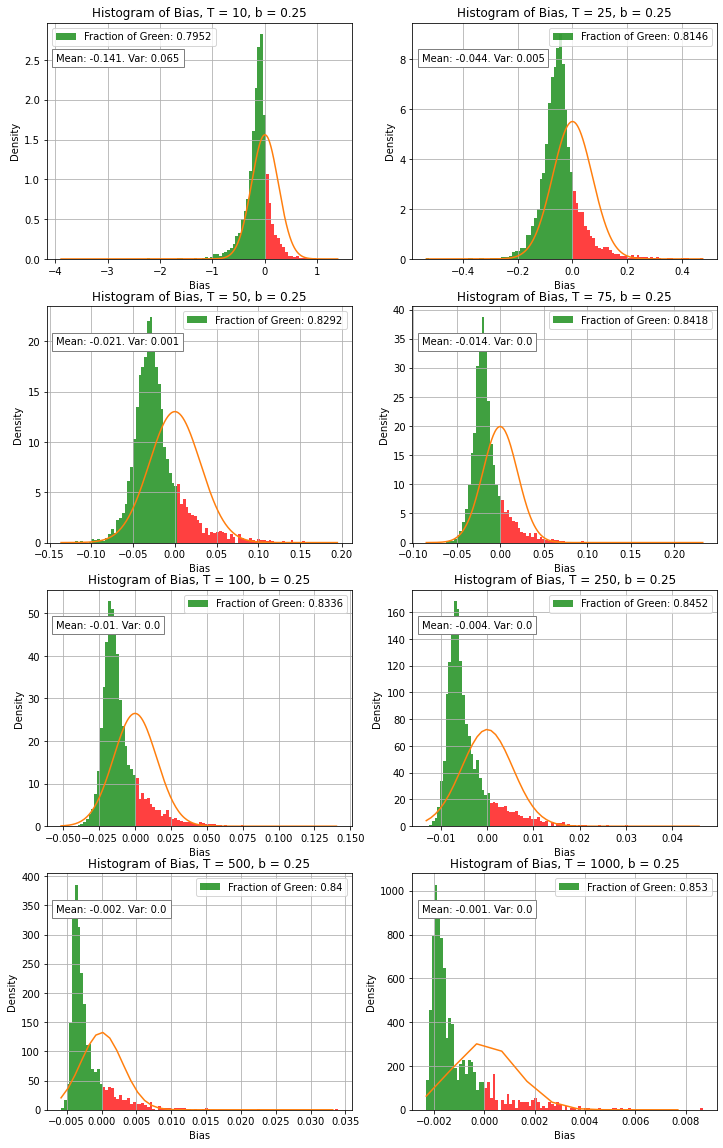


Interestingly, it barely matters for a larger b, up to and until we get very close to a random walk. We also see that, apart from *b = 0.99*, the increase for all four values as a function of T remains equal. A larger *b* has a slightly larger probability of success, but very marginal.

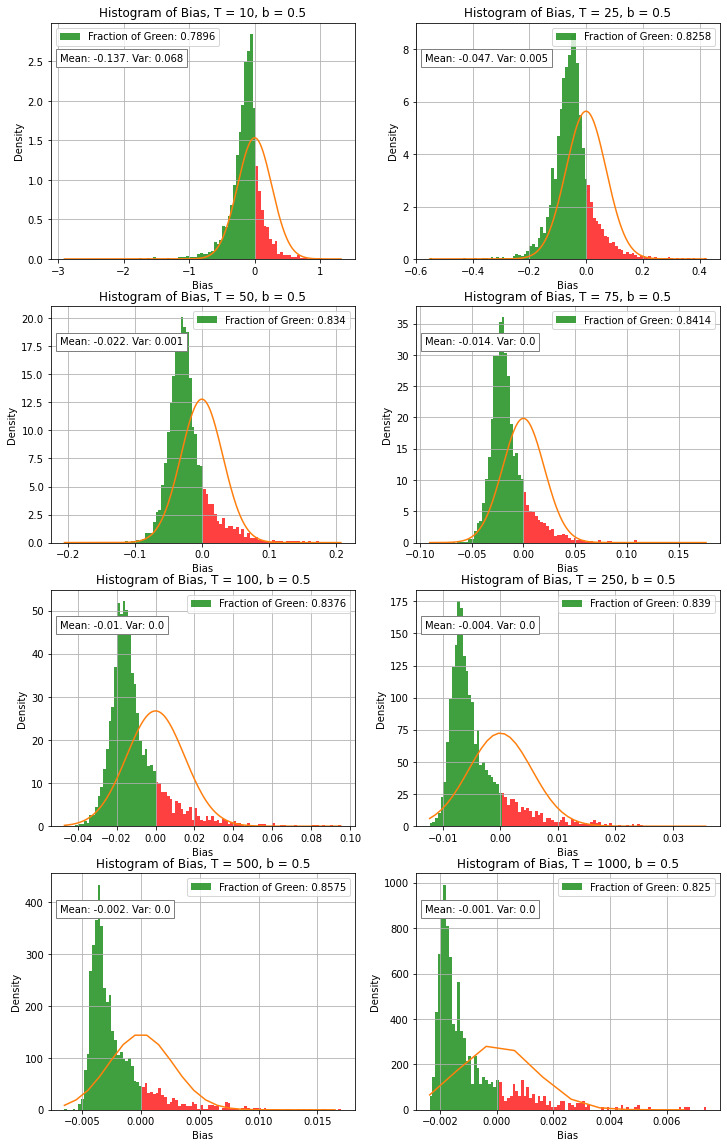
## Bias Histogram, *b = 0, so X2 is Gaussian White Noise*



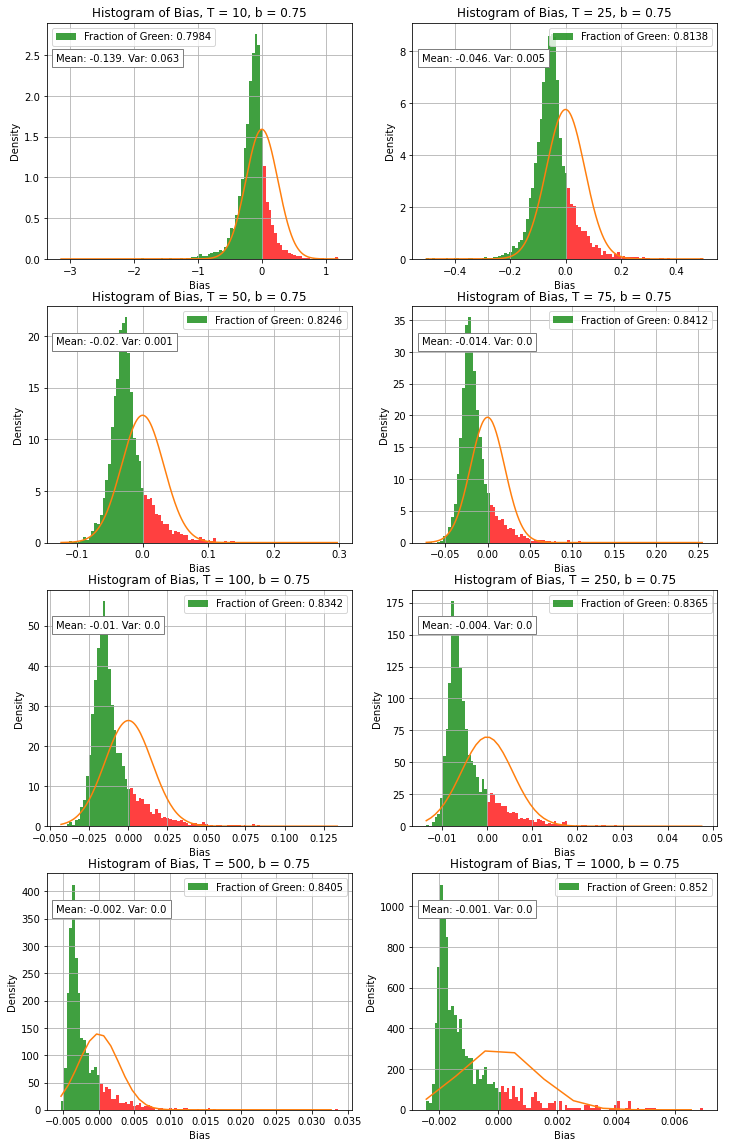
## Bias Histogram, *b = 0.25, so X2 has small dependency*



## Bias Histogram, *b = 0.50, so X2 has small dependency*

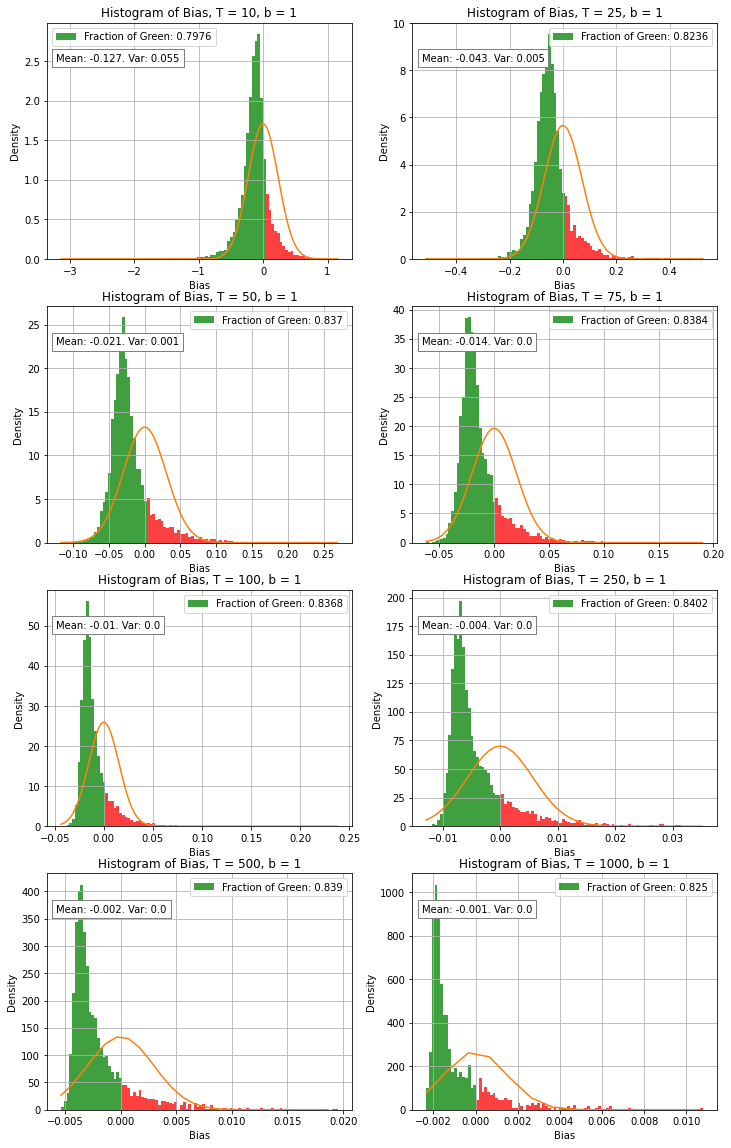


## Bias Histogram, b = 0.75 so heavy dependency

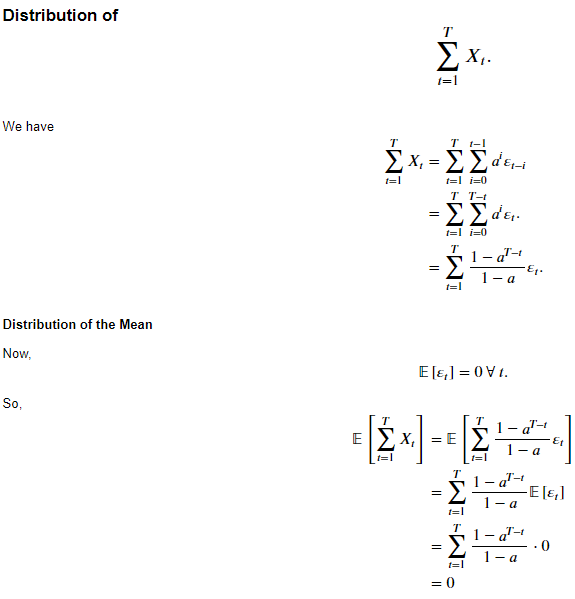
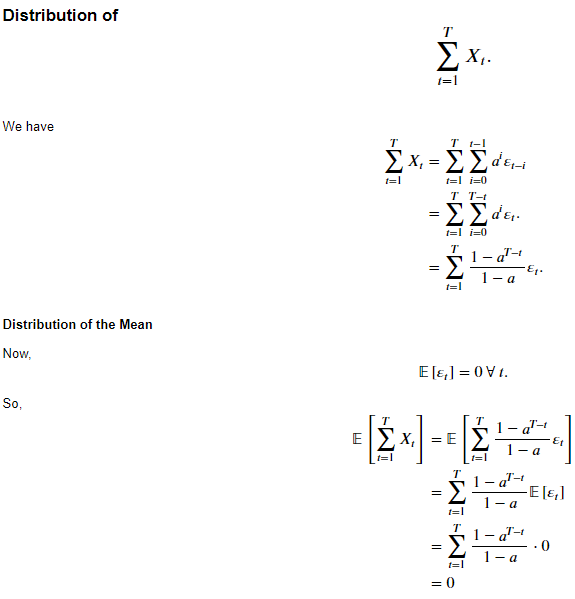


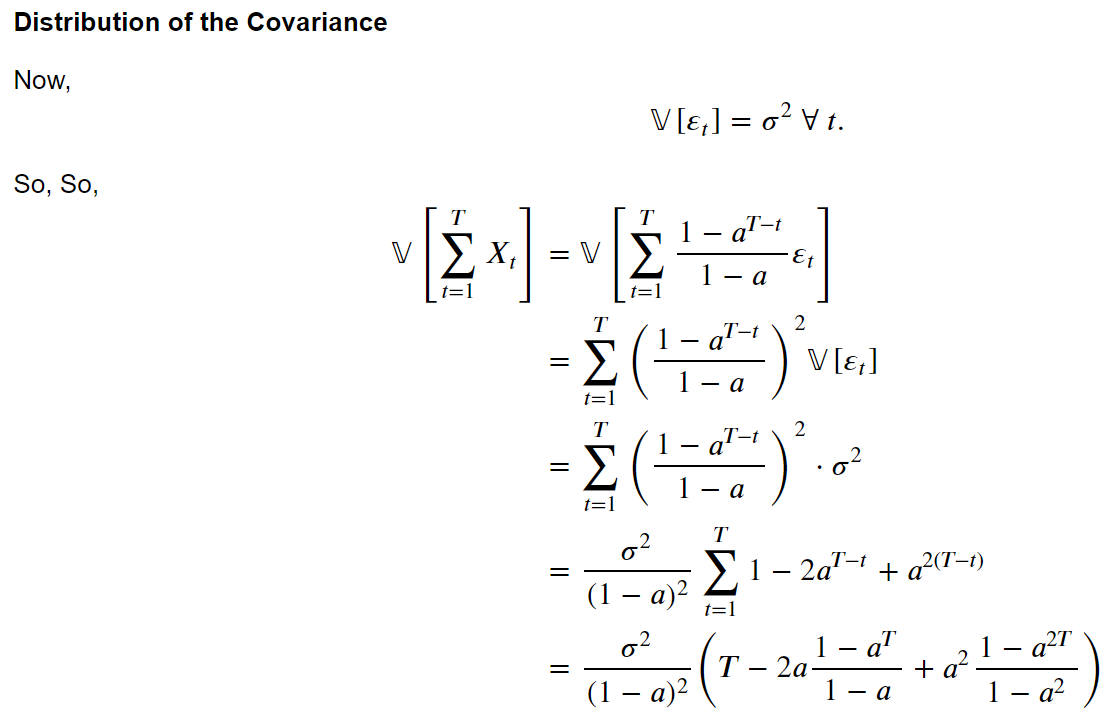
## Bias Histogram, b = 0.99

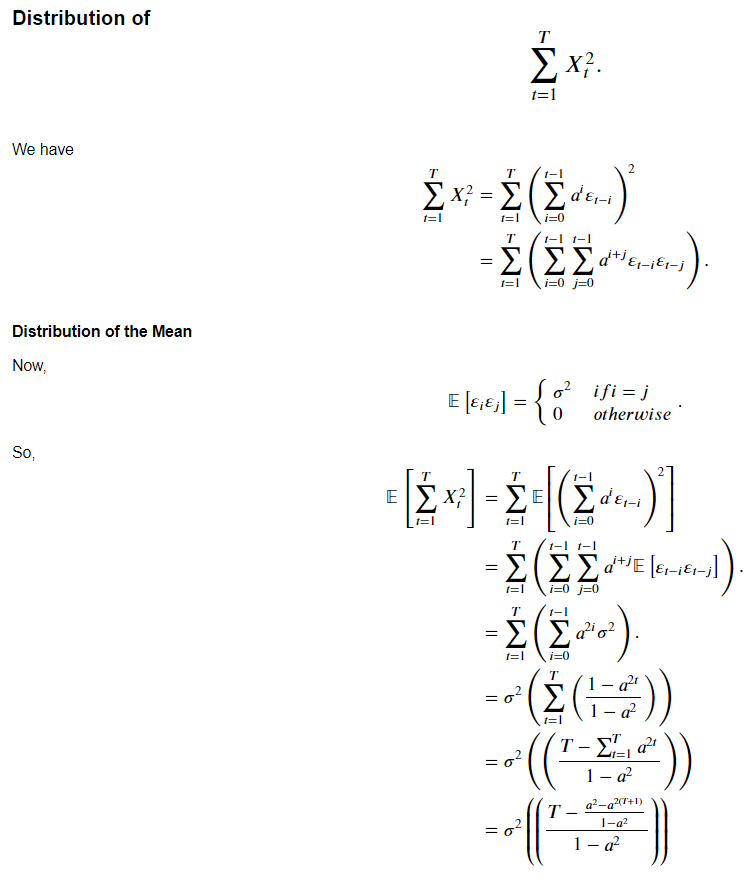
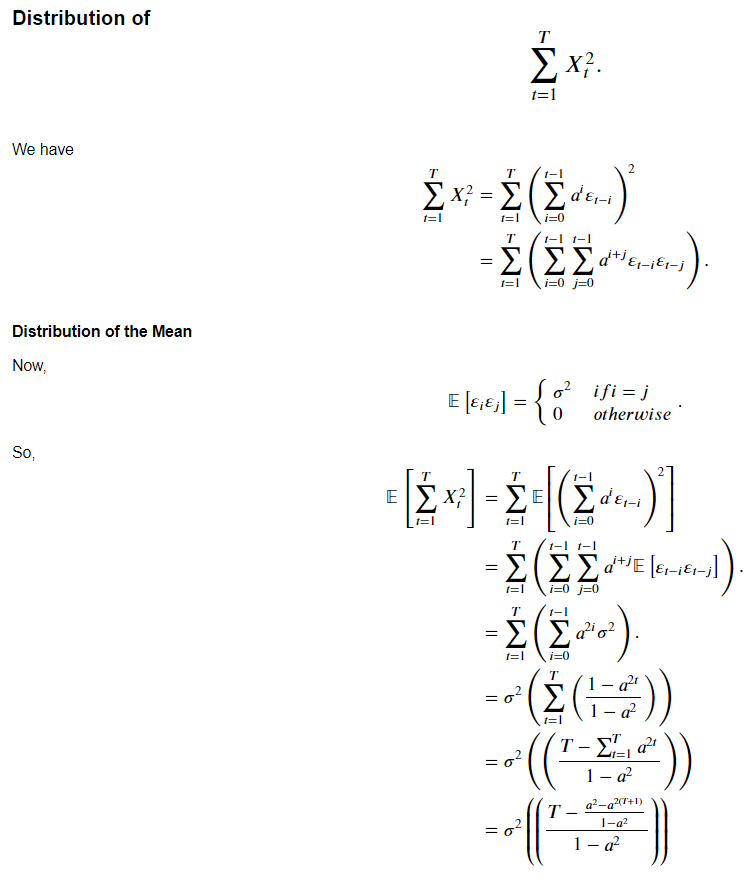
## Bias Histogram, b = 1.00, Random Walk, Initial point = 0.



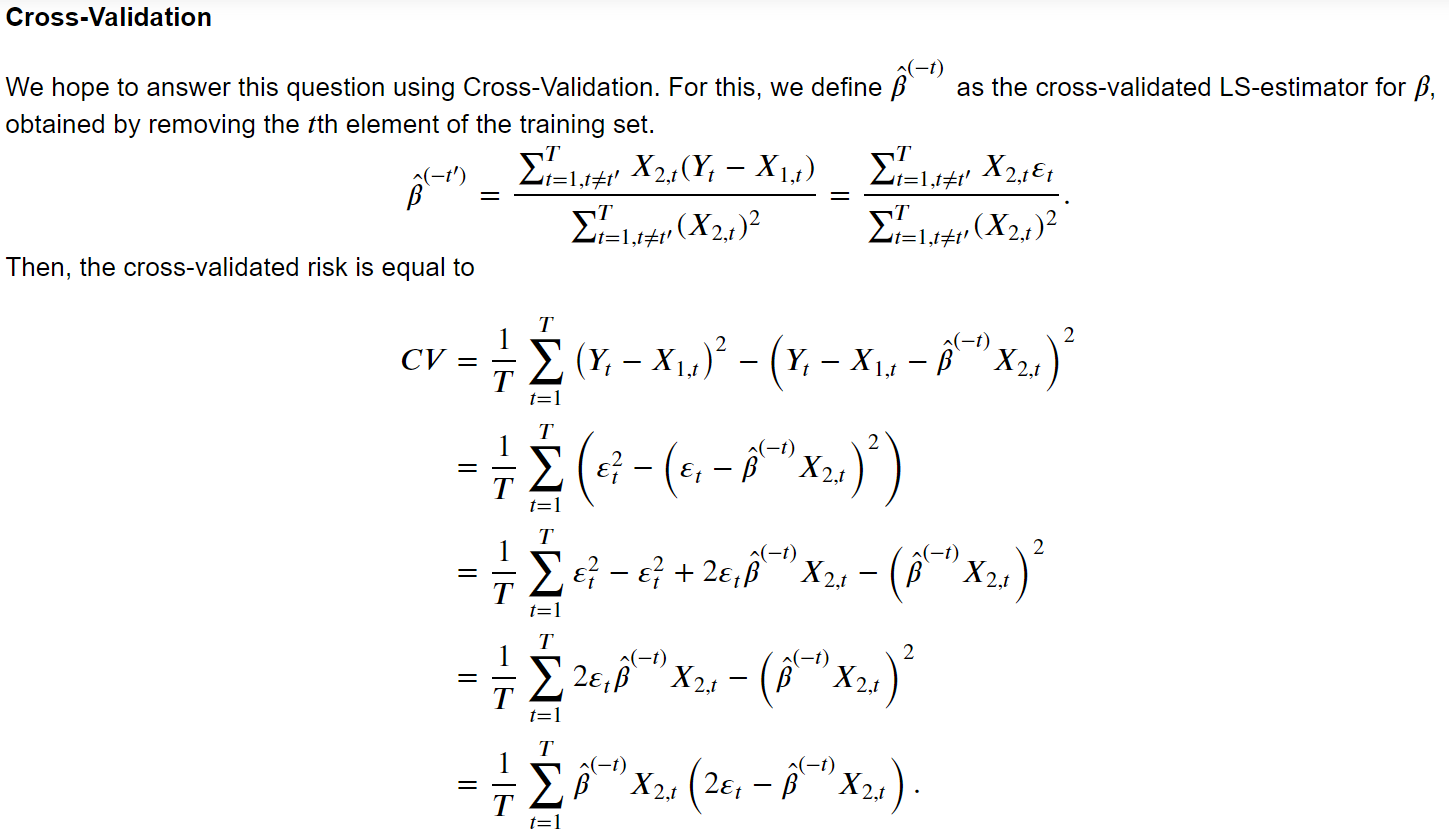
# Distribution of Sum and Sum of Squares for AR(1)

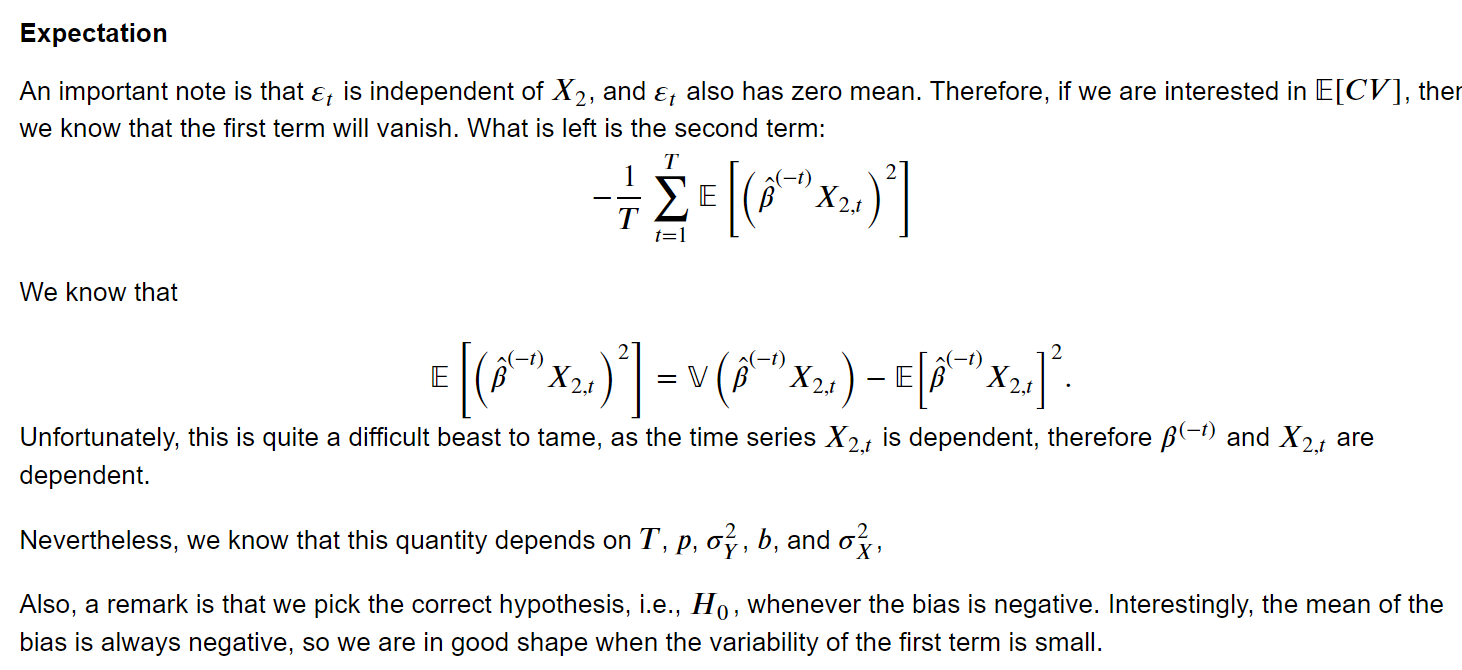


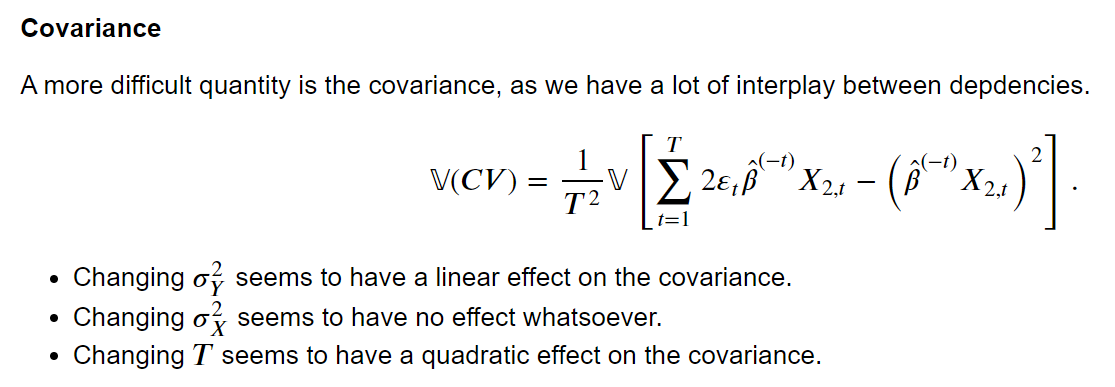


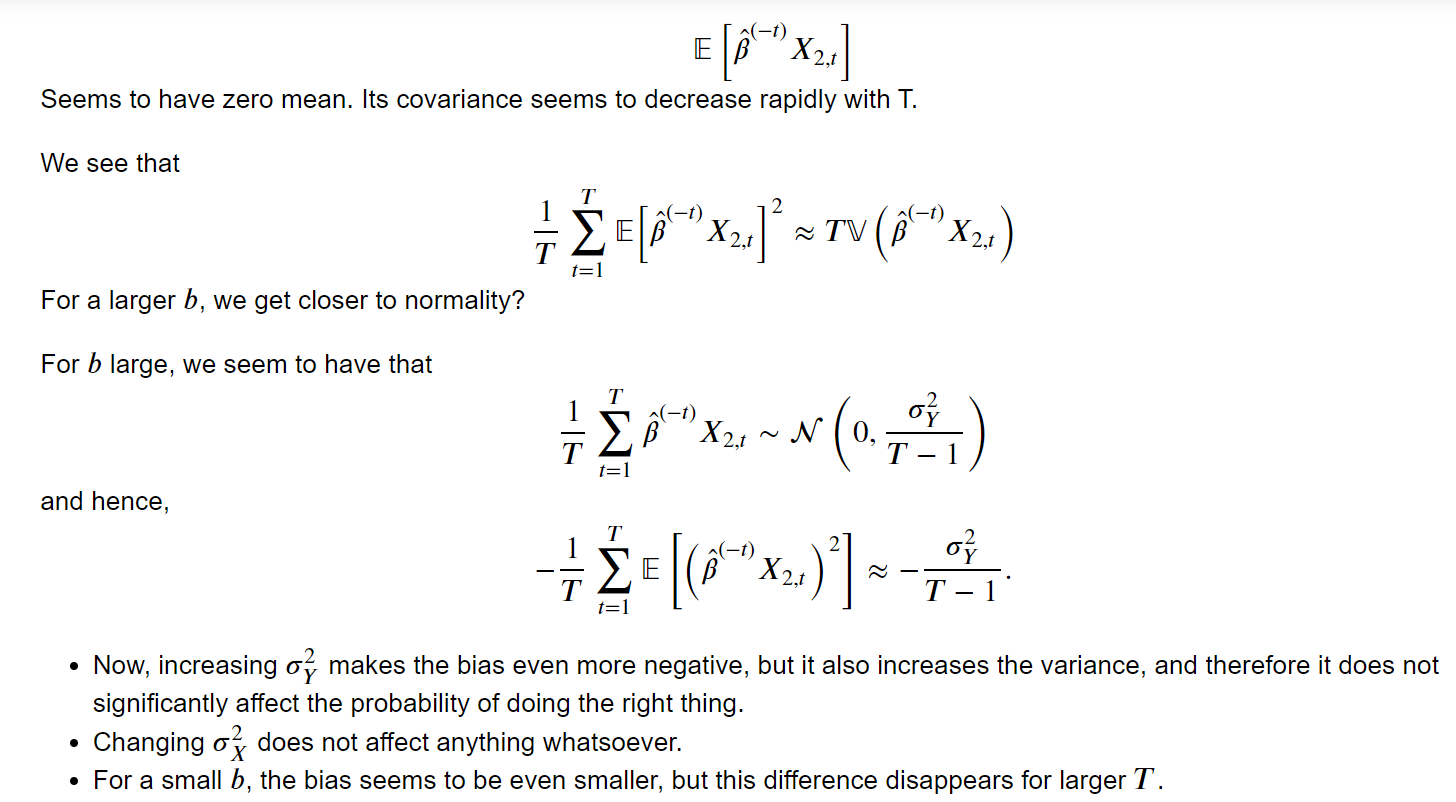


# Distribution of CV Estimator



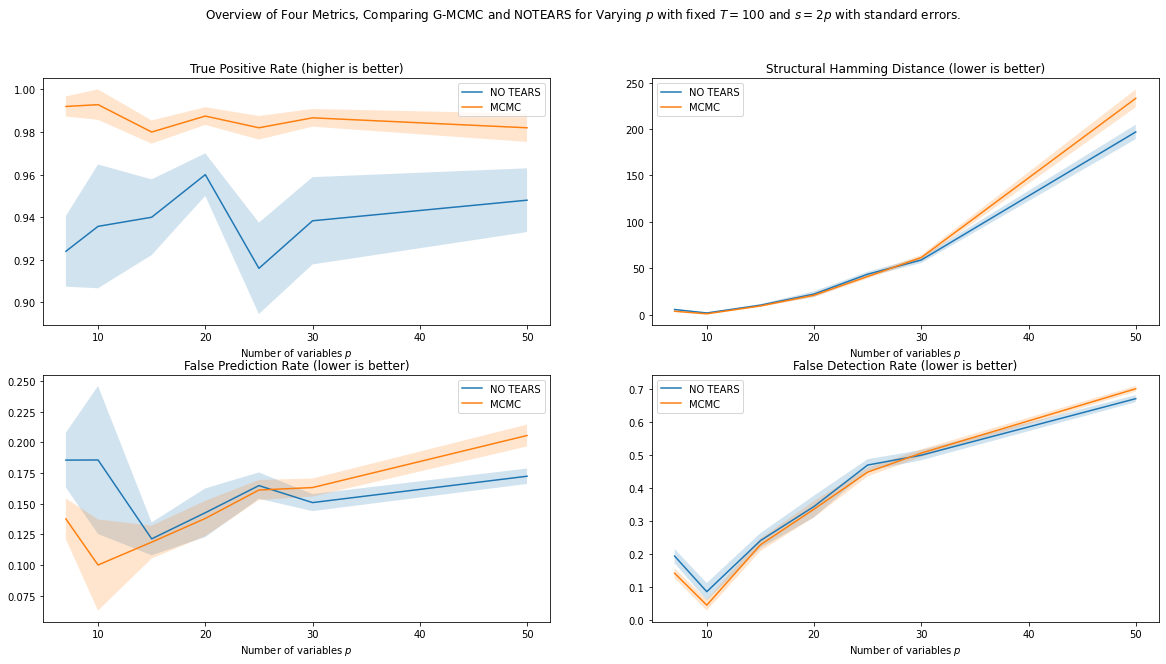






# Greedy MCMC On SEM data

Revisited the Metropolis-Hastings algorithm, and tweaked the probability of acceptance. Now, it is a greedy probability of acceptance, which accepts a transposition if and only if the corresponding DAG solution has a lower likelihood. It seems to work quite well. We run it for as long as NO TEARS takes to produce an output. Up to and until 50 dimensions, it seems to be equally good, but we see the drop-off happening already.



We indeed see that for small *p*, there are less permutations to try, and we can try more permutations per second. Hence, G-MCMC is still quite good here. Nevertheless, as the number of variables *p* grows, the number of permutations grows as *p!*, and the amount of time to try one permutation is approximately *O(p^2)*? It is unlikely that NO TEARS scales poorer, and we indeed see that NO TEARS catches up on G-MCMC at ~30 variables.

# Presentation

Need to do a ~20 minute presentation at SIOUX for my application. Decided to do it about the graduation project. To focus on the story-telling, I think I will stick with structure learning and discuss the OMP, NO TEARS, and MCMC.

1. Introduce the problem, learning a graphical model from data.
2. Applications, e.g. root cause analysis, systems verification.
3. Methodologies:
   1. Combinatorial: Use permutation matrix + OLS.
   2. Continuous: NO TEARS, also as benchmark.
   3. Greedy: Greedy-Metropolis Hastings.
4. Experiments
5. Conclusion
6. Future Directions.

I can send a Teams Invite, or forward the presentation to you for feedback / if you are interested.