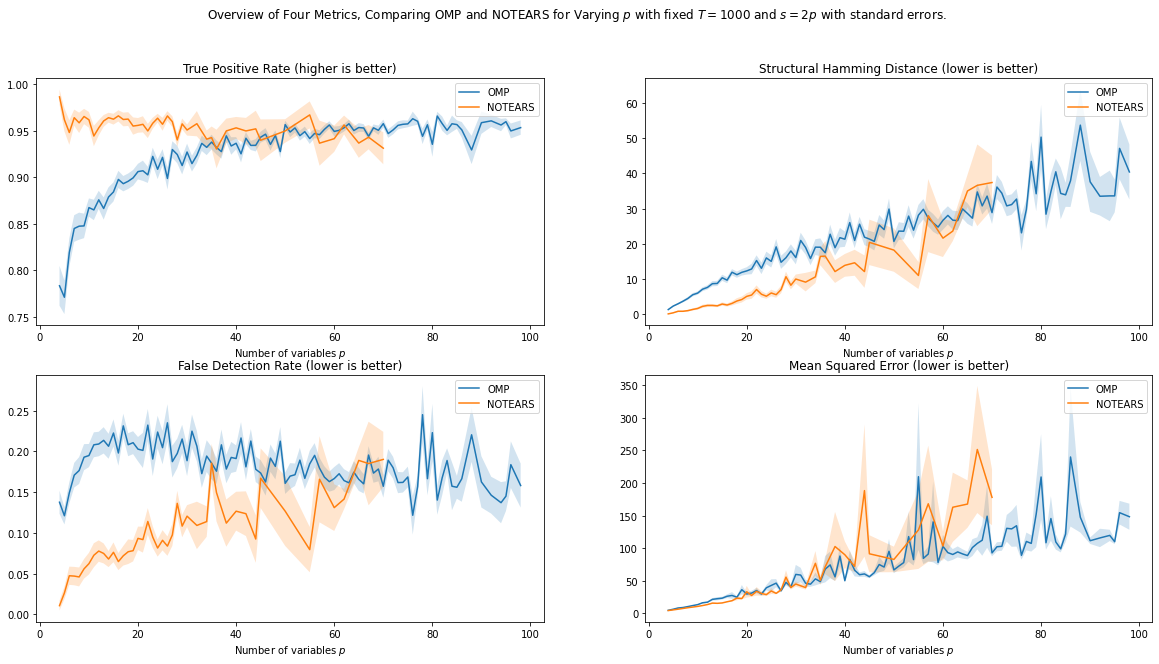
Prep Meeting 22

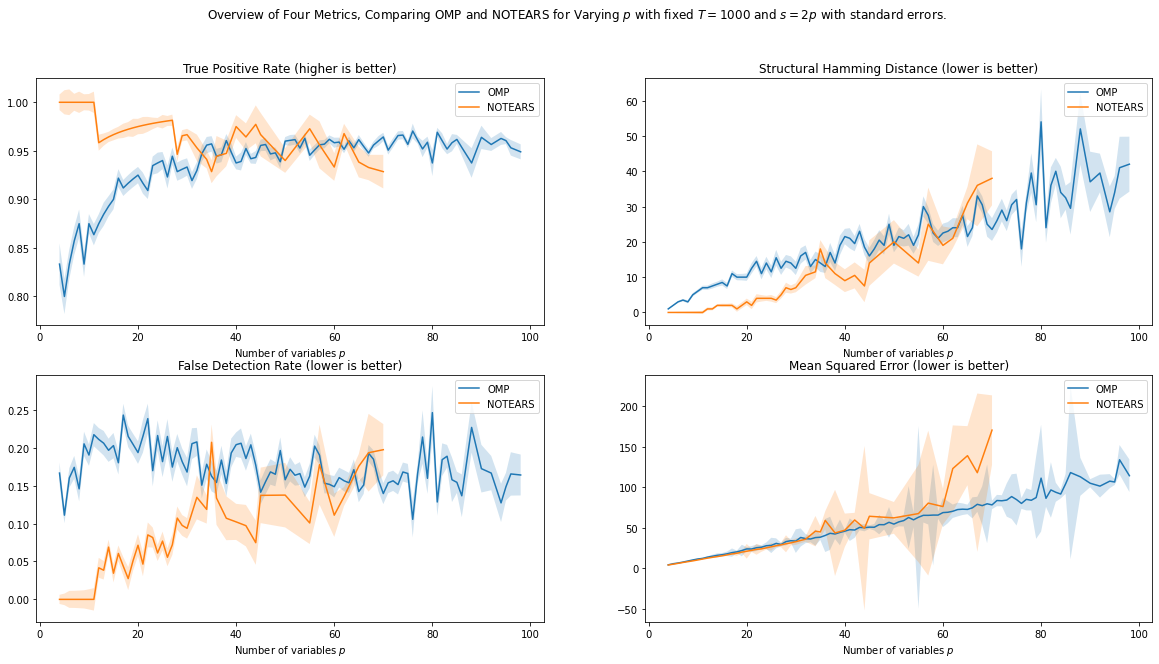
# SEM (Continued).

Conclusions: OMP keeps on being about 100 times faster, but it can be much faster! “Regularization” by thresholding is only done after we have a dense graph, requiring p (p - 1) / 2 iterations first. However, using even some simple thresholding such as 1e-2, I expect to often we done much earlier, but I decided to not change this halfway through.

**Means**



**Median**



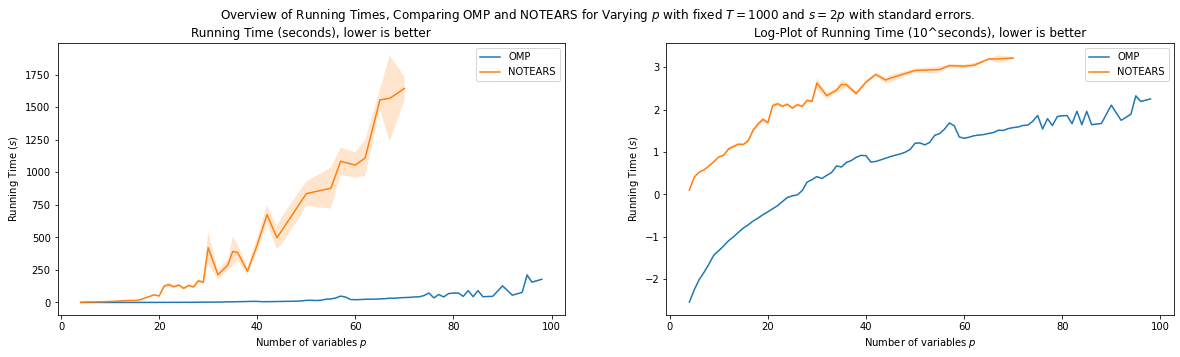
The results are promising! OMP seems faster, and performs better for larger dimensions even. The results for NOTEARS are also in the same ball-park as their plots, so I have confidence they are comparable.

Notes:

* Results are less “stable” at the end, most likely due to less samples (5 rather than 50).

*Follow-Up:*

* Do some more experiments to get more confident results, up to 100 dimensions, further seems quite impossible for NOTEARS.
* Try different sparsity levels (now we have 2p edges per variable, but NOTEARS also tried 4p).
* Try different graphs (NOTEARS also tried scale-free graphs).
* Different additive noise distributions (NOTEARS also tried Exp, and Gumbel). Could also try uniform noise, etc.
* Compare other methods? NOTEARS claims to outperform other methods, but perhaps it is also interesting to include other methods? E.g. LiNGAM, although that one explicitly says that noise is *not* Gaussian.
* Different regularizer for OMP. For now, we just did edge threshold, but perhaps we can try something different? I expect this regularizer to achieve very good FPR and TPR, since the regularizer explots the knowledge that our coefficients are > 0.5.



**Speeding up OMP**

*QR Decomposition using Modified Gram-Schmidt:* Still need to invert p x p matrix, which can be costly in high dimensions. Read a paper that talks about doing a QR decomposition to compute the pseudo inverse or information matrix. This can be done best using a modified Gram-Schmidt procedure. I need to investigate how fast this is, and whether this works with an already “summarized” matrix Psi, where we only store inner products.

*Checking for DAG-ness*. Checking for DAG-ness is not very fast now. I checked it and sped it up a bit. Although it was naïve, not very much gain here, as it is O(p^2). Now two methods for checking DAG-ness, one method has three methods.

*Early stopping*. We do not need to continue until we have a full DAG. Especially for large dimensions, a large portion of coefficients will have a very small gain, so when all edges have a gain below tol, we can stop. I tried putting this at 0.5, and it seems we have gained quite some time while not losing any accuracy.

# Regularizations for VAR data

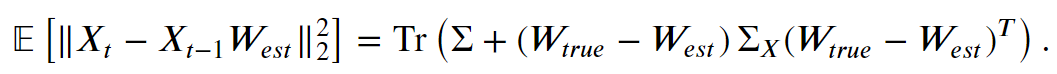
**Formalized Some Regularization Methodologies.**

## Regularization Performance Measures

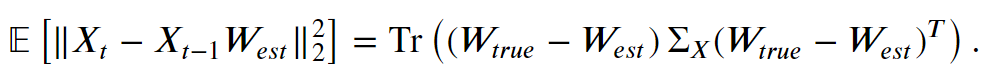
### Predictive Performance Measures

**Mean Squared Error on Test data**.

**Mean Squared Error in Population Setting**

Some time ago, we derived that when data is generated by W\_true, the expected mean squared error using W\_est is equal to, 

Where Sigma\_X is the unconditioned covariance of the time series, and Sigma is the covariance of the additive noise at each time step.

We also know that W\_true is the unique global minimizer of the mean squared error. So we can also subtract this quantity and simply focus on

### Structural Performance Measures

**Area Under Curve**.

Ideally, we want a regularization procedure that works as follows:

* As we increase the degree of regularization, we prevent **overfitting**. We hope to first filter out untrue edges. This will decrease that false positive rate, and leave the true positive rate untouched. We hope to find a point where we have filtered out *all* untrue edges (so FPR = 0), yet we have not filtered out *any* true edges (so TRP stays maximal).
* If we continue to increase the degree of regularization, then we will regularize too much, we **underfit**. We will then also start filtering out true edges, thereby decreasing the TPR.

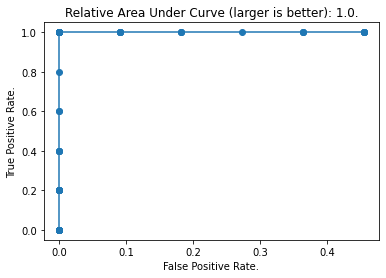
For each threshold value tau, we get a FPR and a TPR. When we plot these pairs (FPR, TPR) as a function of tau, we get what is called an ROC-curve. Such an example is given below.

[[ 0.17 0. 0. 0. ]

[ **0.5** 0.05 0. 0. ]

[ **0.46** -0.09 **-0.57** 0. ]

[-0.12 **-0.43** **-0.44** 0.07]]]



We see that for a threshold of 0, we have a large TPR, but also a large FPR. Then, as we start regularizing, we see that we are doing a good job! We first remove the untrue edges, thereby decreasing the FPR without decreasing the TPR. Then, at the top-left corner. We have done a perfect job, we have found a threshold such that the TPR = 1.0, and the FPR = 0.0.

Afterwards, when we continue to regularize, we see that we start filtering out true edges as well, thereby decreasing the TPR.

The difficult key is knowing when we have reached a good threshold, but this graph can be useful in comparing different methods when we have a ground truth. Such a quantity can for example be the Area Under Curve, which is also used often in machine learning, albeit in a different setting.

Regularization methods such as these:

* Coefficient size.
* Increase in MSE per edge.

# Midterm Proposal

Wrote down the midterm proposal, beginning was quite retrospective.

# Proof into OMP for SEM

Been gathering some sources for guarantees on the OMP with noise. There are a lot if papers with names like “exact recovery conditions for OMP”.

OMP without noise:

OMP with noise:

<https://math.mit.edu/~liewang/OMP.pdf>

<https://arxiv.org/pdf/1905.12347.pdf>