I don't think this can really help anything or at least I don't see it. Not sure if using nuclear norm instead of the **UL** decomposition would help. Perhaps yes ... *Todo: Look at nuclear norm minimisation*

3 Restrict directly (partial-)covariance instead of W (last updated 2/5/2015)

Yule-Walker equations? Does this even make sense? What does it really mean "constraining the covariances/partial covariances"?

The original definition of Granger-causality was for 2 variables (or 2 n-dimensional processes) only. That is can I improve prediction of z_t given the past of x_t? I'm playing with many variables - the problem here is that some may improve prediction of z_t by passing through another variable. Eg y_t may influence x_t and this in turn z_t. How is this treated in the model, the Granger graphs and what would I like to see?

3.1 GPs again

Aha?! So .. what if I formulate the problem as Yule-Walker but instead of putting constraints on norm in $\mathbf{W}$ I directly put some constraints on the covariance estimates?

Perhaps could be more obvious through Bayesian priors? Prior for $\gamma_{ij}(h) \sim \mathcal{N}(0, \sigma)$. Is this in fact a hyperprior on the covariance matrix of the original multi-variate Gaussian process (Gaussian random field)?

In fact, given that the covariance in GP and kernel in the kernel learning theory coincide, imposing sparsity constraints on the covariance somehow relates to Francesco's problem of learning sparse output kernel.

I can look at the vector prediction problem in VAR as at a scalar prediction problem where the specification of the task l is an input for the prediction function.

The following is mainly based on [4] and [5] In the multiple time series prediction problem we have got a data sample $\{y_{i=t\times l} : t \in \mathbb{N}_T, l \in \mathbb{N}_m, i \in \mathbb{N}_{Tm}\}$ which we consider to be a realization of a real Gaussian process $f(.) \sim GP(\mu(.), k(.,.))$ with mean function $\mu(.) = E[f(.)]$ and covariance function $k(.,.) = E[(f(.) - \mu(.))'(f(.) - \mu(.))]$ taking as inputs the 2-dimensional vectors $[t, l]_i$ of all possible time and task combinations (GP is a distribution over function $f : \mathbb{N}_T \times \mathbb{N}_m \to \mathbb{R}$).

We will look at $f(.)$ as at an infinite dimensional vector (whose distribution is given by the GP) but in fact we're interested only in the Tm long vector corresponding to our sample (or its parts) and its joint probability distribution which is a multivariate Gussian distribution (by the marginalisation property of Gaussians) $\mathbf{y} \sim N(\mu, \mathbf{K})$, $\mu_i = \mu([t, l]_i)$, $K(i, j) = k_\xi([t, l]_i, [t, l]_j), \forall i, j \in \mathbb{N}_i$. Here $\mu(.) : \mathbb{N}^2 \to \mathbb{R}$ is the mean function, $k_\xi(.,.) : \mathbb{N}^2 \times \mathbb{N}^2 \to \mathbb{R}$ is the covariance function (or kernel) with hyper-parameters ξ, and the inputs are the time/task indeces $\{[t, l]_i : i \in \mathbb{N}_{Tm}\}$.

Observe that this formulation of the covariance function k is very similar *Note: identical?* to the $\mathcal{Y}$-valued kernel formulation $H(x, x')_{rc} = H((x, r)(x', c))$ where x's correspond to the time indeces t and r, c to the task indeces l.

$$\begin{aligned} y_i &= f([t, l]_i) + e_i, \quad \forall i \in \{1 \dots Tm\} \\ f &\sim GP(\mu, k), \quad e_i \sim N(0, \sigma^2) \end{aligned} \tag{16}$$

[4] actually builds the GP theory and then goes onto Yule-Walker as well but only in the one-dimensional case, he does not go into any details for the multi-dimensional case.

3.2 Shape of kernels 12/5/2015 - NEW!

There are four general types of problems I may want to explore here (of which the first I mainly use as a building stone for the others):

AR 1-step Single time series prediction for 1-step ahead where the output is a scalar

AR h-steps Single time series prediction for h-steps ahead where the h-steps $\{1, \ldots, h\}$ are predicted simultaneously as an h-long output vector

VAR 1-step Multiple time series prediction for 1-step ahead where all the k series are predicted simultaneously as a k-long output vector

VAR h-step Multiple time series prediction for h-steps ahead where the h-steps for all the k series are predicted simultaneously as a $k \times h$ output matrix

Some thoughts about the shape of the kernels:

- For a stationary AR the gram matrix (over $\Delta(t)$) shall be a Toeplitz matrix with diminishing elements. In this way, I know the kernel values for a new observation since it is just an extension of the Toeplitz matrix.
- I could bring some non-stationarity into this by for example making the Toeplitz diagonals some smooth functions
- For s stationary AR with h-steps ahead the input and output kernels (over $\Delta(t)$) shall be identical. I guess, this is the reason why one can show that the optimal h-step forecasts is from the 1-step recursions.
- Full stationarity is unrealistic - extend the smooth functions on the Toeplitz diagonals from the input kernel to the corresponding diagonals of the output kernel?
- Does the separation of the Y-valued kernel into input and output even make sense here?
- For VAR 1-step ahead I could work with separable Y-valued kernel so that the input kernel has the same properties as an AR kernel (e.g. Toeplitz matrix over $\Delta(t)$) etc.) and I need to focus on the output kernel.
- The output kernel works over the task indexes where distances or ordering does not make sense so it cannot be a function of $|k - l|$ (or can it?).
- A straightforward assumption corresponding to my FVAR methods is that L is low-rank and sparse (I think).
- But in fact, L probably should not be symmetrical because the Granger links are not.
- The assumption used e.g. in [2] that the output kernel L is constant across all the input instances is perhaps too strong - the relations between the outputs may also evolve and change with time.
- This suggests that the separation into input and output kernel may not be the best idea here.

3.3 Ideas from group meeting 4/5/2015 - NEW!

- work with single time series but for multi-step ahead prediction
- similarity between input and output kernel via the fact that kernel $= XX'$ and covariance $= X'X$ are linked by the eigen-decomposition (share the same eigenvalues etc.)