Turbo Reconstruction of Structured Sparse Signals

Phil Schniter

THE OHIO STATE UNIVERSITY

(With support from the Office of Naval Research)

CISS, March 2010

The sparse reconstruction problem:

Given measurements

$$\boldsymbol{y} \;=\; \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w},$$

where

$$\begin{aligned} \boldsymbol{A} \in \mathbb{C}^{M\times N} &\quad \text{is a known matrix and} \\ \boldsymbol{w} \in \mathbb{C}^{M} &\quad \text{is CAWGN,} \end{aligned}$$

we want to estimate

$$\boldsymbol{x} \in \mathbb{C}^{N} \quad \text{with at most } K \text{ non-zero elements.}$$

where $K < M < N$.

The sparse reconstruction problem (cont.):

When $\boldsymbol{A}$ has sufficiently incoherent columns, can accurately reconstruct sparse $\boldsymbol{x}$. In particular,

- If $\boldsymbol{A}$ satisfies the $2K$-RIP:

$$\exists \delta_{2K} \text{ s.t. } (1-\delta_{2K})\|\boldsymbol{x}\|_2^2 \leq \|\boldsymbol{A}\boldsymbol{x}\|_2^2 \leq (1+\delta_{2K})\|\boldsymbol{x}\|_2^2 \quad \forall\ 2K\text{-sparse } \boldsymbol{x},$$

  then several approaches yield estimates $\hat{\boldsymbol{x}}$ of K-sparse $\boldsymbol{x}$ that satisfy

$$\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|_2 < C\|\boldsymbol{w}\|_2 \quad \text{for some } C,$$

  including

  - convex optimization (e.g., LASSO/basis-pursuit denoising) . . . *fast*
  - matching pursuit (CoSaMP, subspace pursuit) . . . *faster*
  - iterative thresholding . . . *fastest*

- When $M \gtrsim K \log(N/K)$, can construct random $\boldsymbol{A}$ satisfying RIP with high probability using, e.g.,
  - i.i.d Gaussian or Rademacher (± 1) elements,
  - randomly selected rows of the N-DFT matrix.

Structured sparsity:

In practice, sparse signals often have structure beyond simple sparsity.

Examples:

- Persistence across scales:

  With wavelet coefficients generated from natural scenes, a large child coefficient usually has a large parent coefficient.

- Clustered difference pixels:

  Small changes to a given scene manifest as small clusters of perturbed pixels.

- Tracking of a sparse process:

  The sparsity pattern at a given time index is a small perturbation of the pattern at a neighboring time.

Models for structured sparsity:

1. **Deterministic**, via union of canonical K-sparse subspaces $\mathcal{X}_m$:

   - Simple-sparse: $\boldsymbol{x} \in \Sigma_K \triangleq \bigcup_{m=1}^{m_K} \mathcal{X}_m$ for $m_K = \binom{N}{K}$
   - Model-sparse: $\boldsymbol{x} \in \mathcal{M}_K \triangleq \bigcup_{m=1}^{m_K} \mathcal{X}_m$ for $m_K < \binom{N}{K}$

     Examples: *tree sparse, block sparse.*

     [Baraniuk, Blumensath, Cevher, Davies, Duarte, Do, Eldar, Hassibi, Hedge, Lu, Stojnic, . . .]

2. **Probabilistic**, via hidden binary indicators $s_n \in \{0, 1\}$:

$$p(x_n|s_n) = s_n q_n(x_n) + (1 - s_n)\delta(x_n) \text{ for some } q_n(\cdot)$$

   - Simple-sparse: $p(s_1, \ldots, s_N) = \prod_{n=1}^{N} p(s_n)$.
   - Structured-sparse: $p(s_1, \ldots, s_N)$ is a generic (non-factorizable) pmf.

     Examples: *Markov chains, Markov trees, Markov random fields.*

     [Baraniuk, Carin, Cevher, Duarte, He, Hedge, Godsill, Ng, Wolf, . . .]

Reconstruction under probabilistically structured sparsity:

1. Markov-chain Monte Carlo (MCMC):
   - Markov random field [Wolfe,Godsill,Ng 2004]
   - Markov tree [He,Carin 2009]

   Drawback: takes a *very* long time for the chain to converge.

2. Methods that iterate matching pursuit with MAP sparsity-pattern detection:
   - Markov tree [Duarte,Wakin,Baraniuk 2008]
   - Markov random field [Cevher,Duarte,Hedge,Baraniuk 2008]

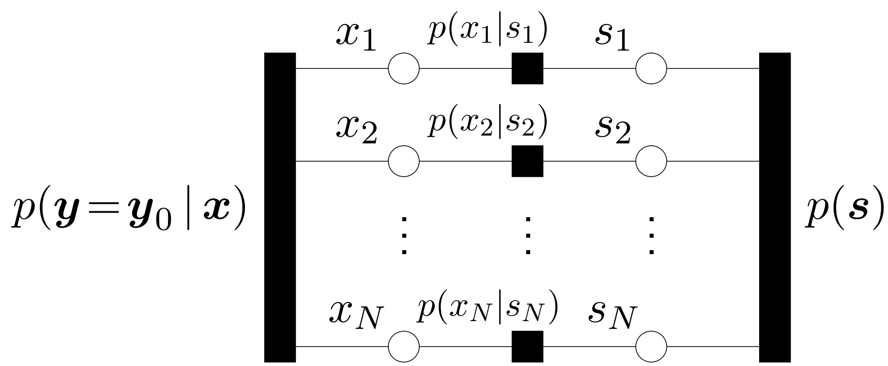
   Drawback: ad hoc.

Belief propagation:

- All inference tasks originate from the joint posterior $p(\boldsymbol{x}, \boldsymbol{s} \,|\, \boldsymbol{y} = \boldsymbol{y}_0)$.
  For example, to infer x_n, we want

$$p(x_n \,|\, \boldsymbol{y} = \boldsymbol{y}_0) = \int_{\boldsymbol{x}_{-n}} \sum_{\boldsymbol{s} \in \{0,1\}^N} p(\boldsymbol{x}, \boldsymbol{s} \,|\, \boldsymbol{y} = \boldsymbol{y}_0)$$

- The **factor graph** of $p(\boldsymbol{x}, \boldsymbol{s} \,|\, \boldsymbol{y} = \boldsymbol{y}_0)$ illustrates statistical dependence among variables:

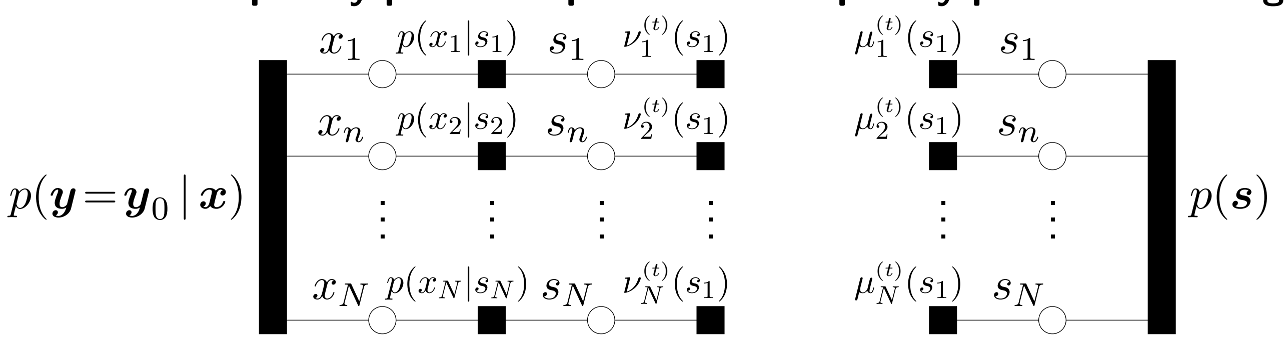


- The factor graph tells us about the complexity of exact inference:
  - no loops: exact inference possible via **message passing**.
  - loopy: exact inference is *NP hard!*

  Even with loops, message passing can yield near-optimal inference with...
  - a few loops: (e.g., turbo decoding, LDPC decoding, inference on MRFs)
  - densely loopy: (e.g., multiuser detection, compressed sensing)

The turbo principle:

- Our (loopy) inference problem can be tackled by *splitting it into two sub-problems and iterating between them*. The t-iteration SPE priors $\{\nu_n^{(t)}\}_{n=1}^N$ are set as the $(t-1)$-iteration SPD output messages, and vice versa.



- This is reminiscent of *noncoherent turbo equalization*:

  Writing $x_n = \theta_n s_n$ for $s_n \in \{0,1\}$ and, say, $\theta_n \sim \mathcal{CN}(0,1)$,
  we get $\boldsymbol{y} = \boldsymbol{A}\,\mathcal{D}(\boldsymbol{\theta})\boldsymbol{s} + \boldsymbol{w}$,
  where we can interpret

  $\boldsymbol{\theta}$ as unknown Rayleigh channel gains,
  $\boldsymbol{A}$ as the known structural component of the channel matrix, and
  $\boldsymbol{s}$ as the coded-bit vector.

Sparsity pattern equalization (SPE):

- For this inference sub-task...
  - The input messages are the extrinsic LLRs generated by SPD, which are converted to pmfs and treated as priors:

$$\lambda_n^{\mathsf{SPE}} \triangleq \frac{1}{1+\exp(-L_n^{\mathsf{SPD}})} \;\rightarrow\; \Pr\{s_n = 1\}.$$

  - The output messages are the extrinsic LLRs

$$L_n^{\mathsf{SPE}} \triangleq \log \frac{p(\boldsymbol{y}=\boldsymbol{y}_0 \,|\, s_n=1)}{p(\boldsymbol{y}=\boldsymbol{y}_0 \,|\, s_n=0)}.$$

- There are many ways to implement SPE (e.g., MCMC, soft matching pursuit, expectation maximization, belief propagation).
- We will apply the **approximate message passing** (AMP) approach from [Donoho, Maleki, Montanari 2010], which
  - enjoys various optimality properties in the large-system limit, and
  - yields a *very* fast iterative soft thresholding algorithm.

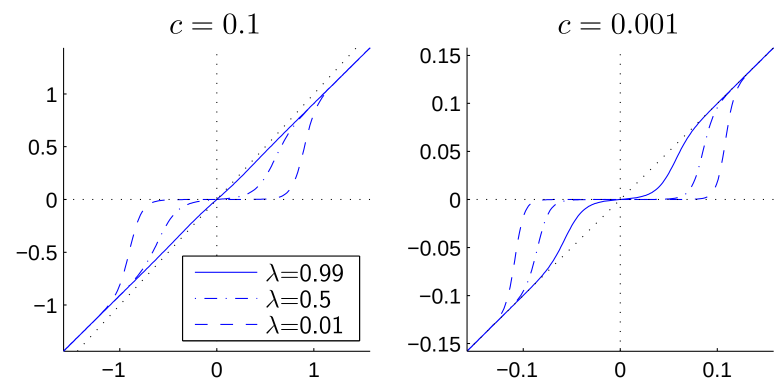
Sparsity pattern equalization (cont.):

The Bayesian AMP algorithm [Donoho, Maleki, Montanari 2010] is:

$$
\begin{aligned}
\boldsymbol{\theta}^i &= \boldsymbol{A}^H \boldsymbol{z}^i + \boldsymbol{\mu}^i && \text{projected residual} \\
\boldsymbol{\mu}^{i+1} &= F(\boldsymbol{\theta}^i; c^i) && \widehat{\mathrm{E}}\{\boldsymbol{x} \,|\, \boldsymbol{y} = \boldsymbol{y}_0\} \\
\boldsymbol{v}^{i+1} &= G(\boldsymbol{\theta}^i; c^i) && \widehat{\mathrm{var}}\{\boldsymbol{x} \,|\, \boldsymbol{y} = \boldsymbol{y}_0\} \\
c^{i+1} &= \sigma^2 + \tfrac{N}{M}\langle \boldsymbol{v}^{i+1} \rangle && \text{effective noise variance} \\
\boldsymbol{z}^{i+1} &= \boldsymbol{y} - \boldsymbol{A}\boldsymbol{\mu}^{i+1} + \tfrac{N}{M}\langle F'(\boldsymbol{\theta}^i; c^i)\rangle \boldsymbol{z}^i && \text{residual}
\end{aligned}
$$

where $F(\cdot, c)$ is a component-wise **soft thresholding function**, $F'(\cdot, c)$ is its component-wise derivative, and $G(\cdot, c)$ is another nonlinear function.

For CWGN $\boldsymbol{w} \sim \mathcal{CN}(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$ and Bernoulli/Gaussian coefficients $x_n \sim \lambda_n \mathcal{CN}(0,1) + (1-\lambda_n)\delta(x_n)$, $F_n(\cdot, c)$ looks like:



The "magic" of AMP is due to the Onsager reaction term $\frac{N}{M}\langle F'(\boldsymbol{\theta}^i; c^i)\rangle \boldsymbol{z}^i$.

Sparsity pattern decoding (SPD):

- For this inference sub-task...
  - The input messages are the extrinsic LLRs generated by SPE, which are converted to marginal pdfs and treated as priors:

$$\lambda_n^{\mathsf{SPD}} \triangleq \frac{1}{1+\exp(-L_n^{\mathsf{SPE}})} \;\rightarrow\; \Pr\{s_n = 1\}.$$

  - The output messages are the extrinsic LLRs

$$L_n^{\mathsf{SPD}} \triangleq \log \frac{\sum_{\boldsymbol{s}_{-n}} p(\boldsymbol{s}_{-n}|s_n=1)\prod_{q\neq n} p(s_q)}{\sum_{\boldsymbol{s}_{-n}} p(\boldsymbol{s}_{-n}|s_n=0)\prod_{q\neq n} p(s_q)}.$$

- We can implement the SPD efficiently using message passing.
  - With Markov-chain or Markov-tree priors, exact inference can be efficiently implemented using the forward-backward algorithm.
  - With Markov-field priors, message passing yields efficient near-optimal inference. [Freeman, Pasztor, Carmichael 2000]

EXIT charts:

- In turbo equalization, **extrinsic information transfer** (EXIT) charts are used to predict how well the equalizer and decoder work *together* to infer the bits.

  In our structured sparse inference framework, an EXIT chart can be used to predict how well SPE and SPD work together to infer the *sparsity pattern*.

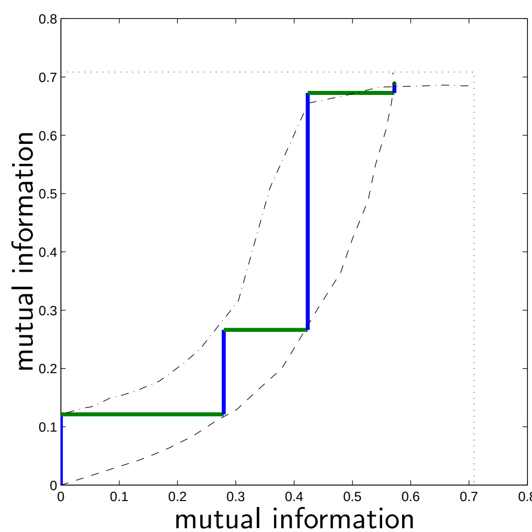
- Say $I^{(t)}_{\text{SPE}}$ denotes the mutual information between the true pattern and the iteration-t SPE extrinsic LLRs, and $I^{(t)}_{\text{SPD}}$ denotes that for the SPD.

  Then the transfer curve $I^{(t)}_{\text{SPD}}$-vs-$I^{(t)}_{\text{SPE}}$ characterizes the decoder, while the transfer curve $I^{(t+1)}_{\text{SPE}}$-vs-$I^{(t)}_{\text{SPD}}$ characterizes the equalizer.

Example for Bernoulli-Gaussian SPE and Markov-chain SPD:

The "EXIT tunnel" accurately predicts the trajectory.

$(\frac{M}{N} = 0.25,\ \frac{K}{M} = 0.8,\ \gamma = 0.2)$:



Performance evaluation:

We evaluated performance empirically using the following data model.

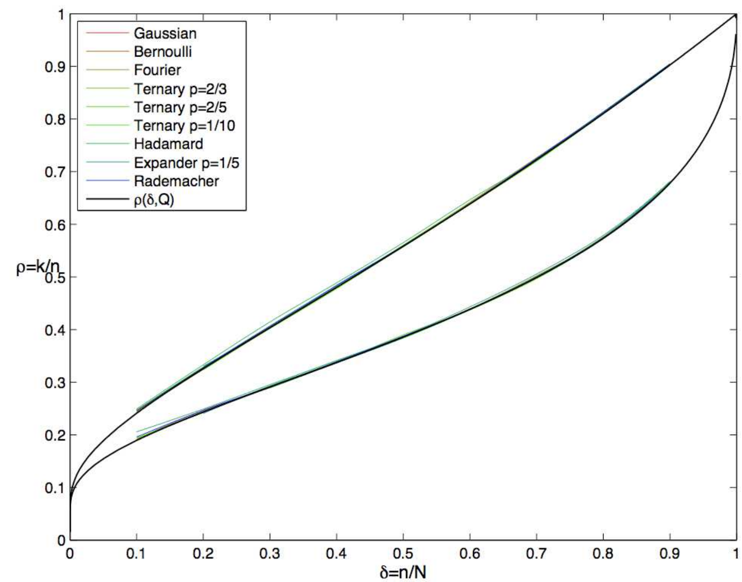
- Elements of $\boldsymbol{A}$ were i.i.d $\mathcal{CN}(0, M^{-1})$.
- Noise samples were i.i.d $\mathcal{CN}(0, \sigma^2)$ with σ^2 so that SNR $\triangleq \frac{\mathrm{E}\{\|\boldsymbol{Ax}\|_2^2\}}{\mathrm{E}\{\|\boldsymbol{w}\|_2^2\}} = 30\text{dB}$.
- Sparsity patterns $\{s_n\}_{n=1}^N$ were generated using a 2-state Markov chain.
  - For a given *sparsity rate* $\lambda \triangleq \Pr\{s_n = 1\} = \frac{\mathrm{E}\{K\}}{N} \in (0, 1]$, can show Markov transition probabilities obey $p_{01} = p_{10}(1/\lambda - 1)$ and $p_{10} = \gamma\lambda$.
  - We refer to $\gamma \in (0, 1]$ as the **independence factor**, since $\gamma = 1$ yields i.i.d $\{s_n\}$ and smaller γ yield $\{s_n\}$ with longer switching times.
- Active signal coefficients x_n were i.i.d $\mathcal{CN}(0, 1)$.

Empirical phase transition curves were calculated based on

$$\text{NMSE} \triangleq \frac{\mathrm{E}\{\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|_2^2\}}{\mathrm{E}\{\|\boldsymbol{x}\|_2^2\}}.$$

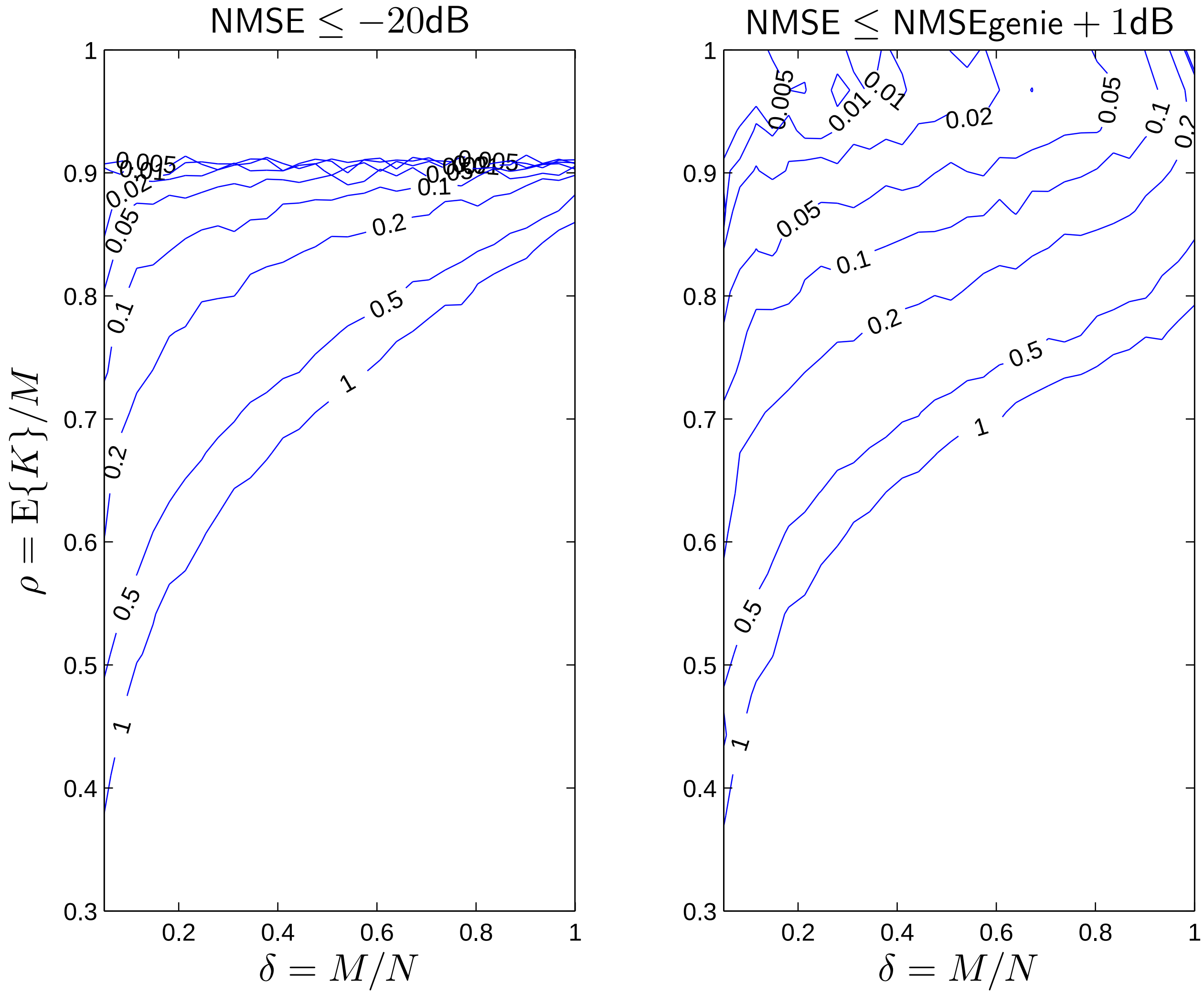
Phase transition curves:

- Consider sparse reconstruction for a given pair (δ, ρ) of **oversampling factor** $\delta \triangleq \frac{M}{N} \in (0, 1]$ and **sparsity factor** $\rho \triangleq \frac{K}{M} \in (0, 1]$.
- In the large-system limit ($K, M, N \to \infty$ for fixed δ, ρ), there often exists a *phase transition curve* that partitions the (δ, ρ) space into regions of *successful* and *unsuccessful* reconstruction.
  - Curves depend on signal class, matrix class, reconstruction algorithm, and definition of success.
  - Have been both empirically observed [Maleki, Donoho 2009] and derived using combinatorial geometry [Donoho, Tanner 2009].
  - Much tighter than corresponding RIP bounds. [Blanchard, Cartis, Tanner, Thompson 2009]



[Donoho, Tanner 2010]

Empirical results for Markov-structured sparsity, for several γ:



900-point (δ, ρ)-grid, 200 realizations, 5 turbo iterations, 10 AMP iterations, $N = 512$ length signal, …only takes a few hours on a desktop computer.

Summary:

- We considered structured-sparse signal reconstruction in AWGN.
- Sparsity pattern structure was modeled using a generic pmf.
- Approximate inference performed via message passing on factor graph.
  - iterates between two soft-input soft-output blocks: SPE and SPD.
  - reminiscent of noncoherent turbo decoding.
- SPE was implemented using the Bayesian AMP method [Donoho, Maleki, Montanari 2010].
- SPD implementation depends on structure of pattern pmf (e.g., we used the forward-backward algorithm for our Markov chain prior).
- EXIT charts were applied to predict interaction between SPE and SPD.
- Empirical phase transition curves were used to quantify NMSE performance relative to (genie-aided) known-pattern MMSE estimates.
- We observed that the phase transition curve moves up and to the left as the sparsity pattern becomes more structured.

Thanks!