Title: Talk 21 via live stream

Speakers: Walter Landry, David Simmons-Duffin

Collection: Bootstrap 2019

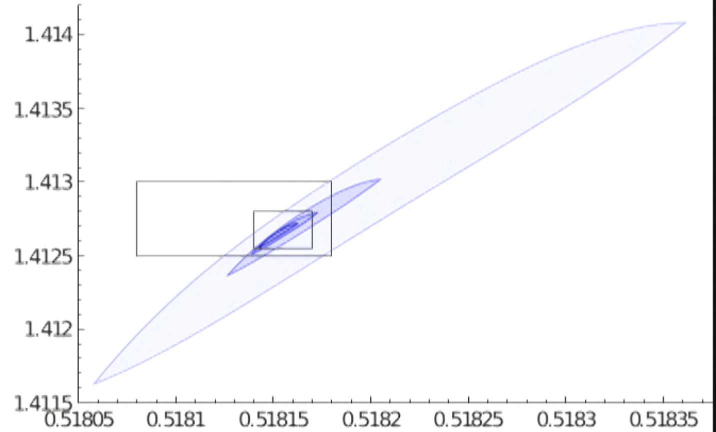
Date: July 29, 2019 - 10:00 AM

URL: http://pirsa.org/19070043

The Bootstrap and Semidefinite Programs

- The conformal bootstrap can be formulated in terms of a semidefinite program.
- Semidefinite programs are generic math problems that occurs in many branchs of science and engineering.
- Existing, off-the shelf solver implementations exist in a variety of environments
  - Matlab, Mathematica, C, Python, ...

Why SDPB?



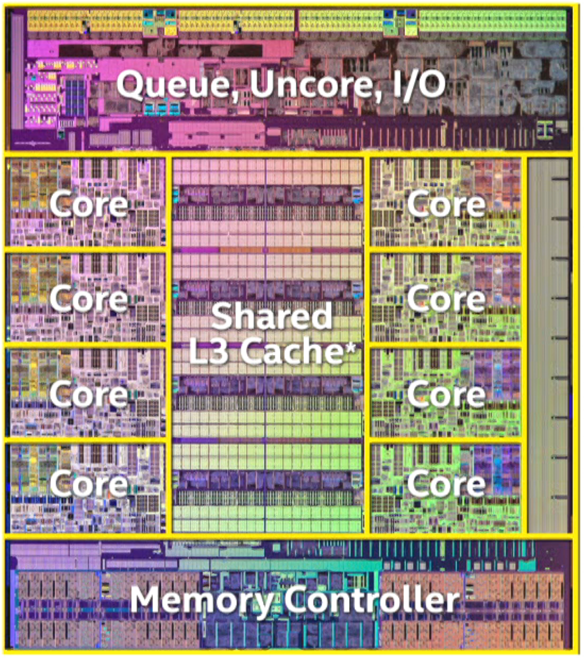
- Bootstrap calculations can require extreme numerical precision and computational resources.
  - Ising computations ran for weeks.
- SDPB is a solver optimized for bootstrapping.
  - Open-source
  - Arbitrary precision
  - Heavily parallelized

Parallelizing Linear Algebra

- Most of what takes time in SDPB is linear algebra operations on independent blocks of matrices.
- The results of these independent operations are combined into a single, comparatively small, global matrix Q.
- These independent operations can be run on different cores.

Parallelization with OpenMP

- SDPB was initially parallelized with OpenMP
  - OpenMP is very easy to use, but it relies upon a global view of memory.
  - Works on single nodes up to ~20 cores.
- Global view of memory quickly stops working beyond a single node.



Parallelism with MPI

- SDPB has been enhanced to use MPI (Message Passing Interface).
- MPI works by passing messages between cores.
- This works far better than OpenMP on supercomputers.
- It required extensive restructuring of the code.

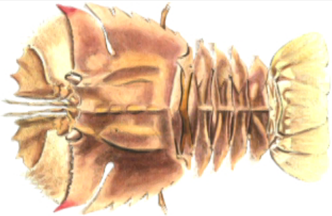
Timing Runs

- Part of the restructuring is that we now have to explicitly assign these block computations to specific cores.

- Doing this well requires measuring how long it takes to run calculations for each block.
- Trying to derive the timings from first principles results in terrible performance.

Recent Work

- Automatic Timing
- Fake Primal Fix
- Faster Input
- Hot-starting and text checkpoints
- Installations
- Memory Use
- Scalar Blocks
- Spectrum Extraction
- Proposals



Fake Primal Solution

- There is a bug in the original, SDPB-OpenMP implementation of the primal error
- Paper says: $primalError = \max\{|\mathrm{p_i}|, |\mathrm{P_i j}|\}$
- Implementation was: $primalError = \max\{|\mathrm{P_i j}|\}$
- **Usually** it makes no difference

  - SDPB-MPI now reports both P and p, but it uses the full primalError for deciding when to stop.

Faster Input

- XML : pvm2sdp
  - Now fully parallelized
- SDP in Mathematica: sdp2input
  - Directly generates SDPB input files.
  - 16 times faster than SDPB.m on 28 cores
  - Enables some people to work without Mathematica (not all clusters have it).

Hot-starting is Fully Supported

- Allows you to start a new calculation with an older solution
- Can reduce the number of iterations by a factor of 10.

Text Checkpoints

- Allows you to add to or modify an existing solution for a new problem.
- Portable across machines
- Not strictly bitwise identical.
  - The last bit can be different.
  - This comes from a limitation in the underlying GMP library.
  - It is unlikely to matter.


PERIMETER
PI
INSTITUTE FOR THEORETICAL PHYS
PERIMETER
INSTITUTE

Easier Installation

- Better autodection of libraries
- No unnecessary dependencies.

Installed Everywhere

HARVARD

LUX ET VERITAS

Caltech

EPFL

XSEDE

SDSC
SAN DIEGO SUPERCOMPUTER CENTER

docker

Easiest Option

- For smaller runs on your laptop or desktop
  - Docker (Windows, Mac, Linux) or
  - Singularity (Linux: recommended)
- Download and Run
- Pretty efficient and uses all cores.
  - IAS admins used Singularity for their install on the Helios cluster.
- Instructions in the repository
  https://github.com/davidsd/sdpb/blob/master/docs/Docker.md
  https://github.com/davidsd/sdpb/blob/master/docs/Singularity.md

Much Better Memory Use

- Memory use is dominated by many cores having their own copy of their contribution to the matrix Q.
- Q is symmetric, so we now explicitly deallocate half of it.
  - The underlying parallel linear algebra library, Elemental, is not accustomed to this, so we have to be a bit careful.

procGranularity

- Added the option procGranularity
- Spreads the local contribution to Q across more cores
- A bit slower, so only use if desperate

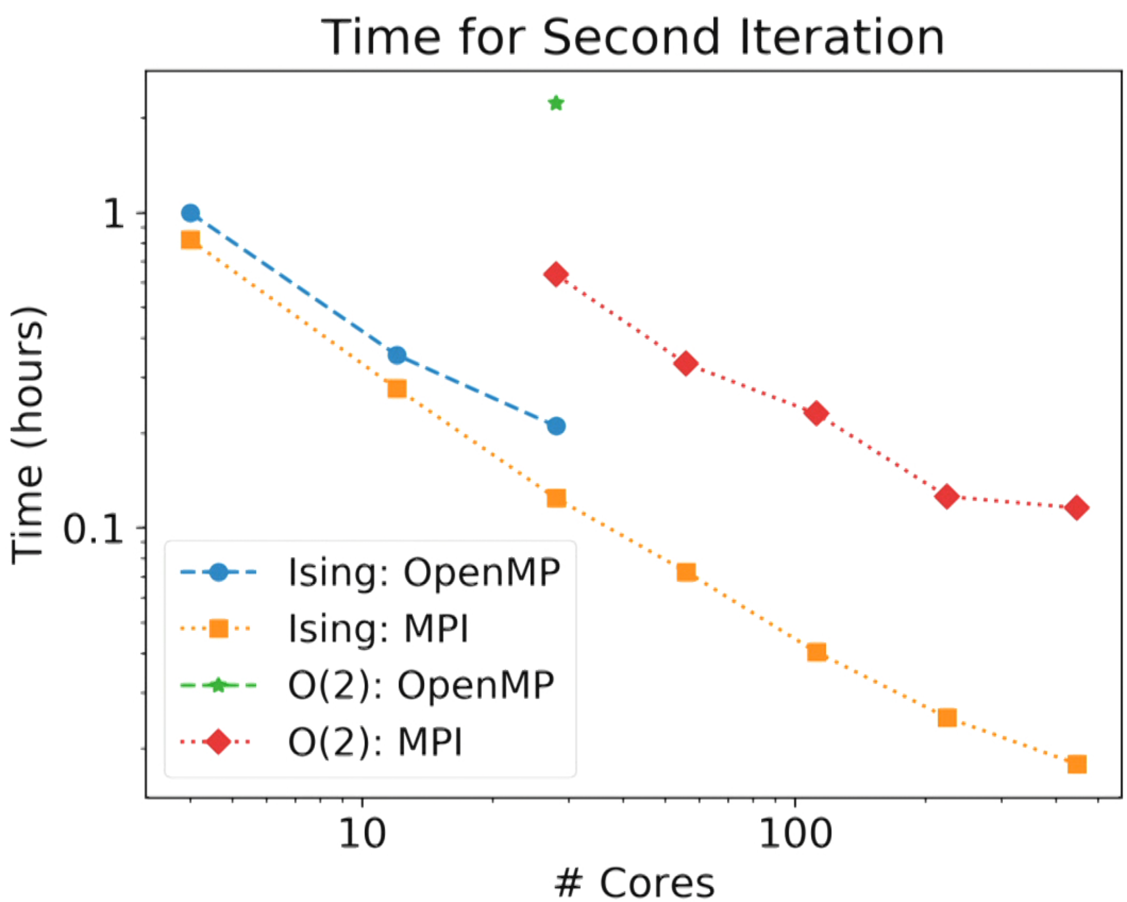
Synchronizing Q

- The local contributions to Q are summed and then distributed to a global Q with the low level routine MPI_Reduce_scatter.
- MPI_Reduce_scatter requires an additional copy of Q on each core.
  - Reimplemented to remove these copies
- With procGranularity, the memory overhead compared to SDPB-OpenMP should now be minimal.

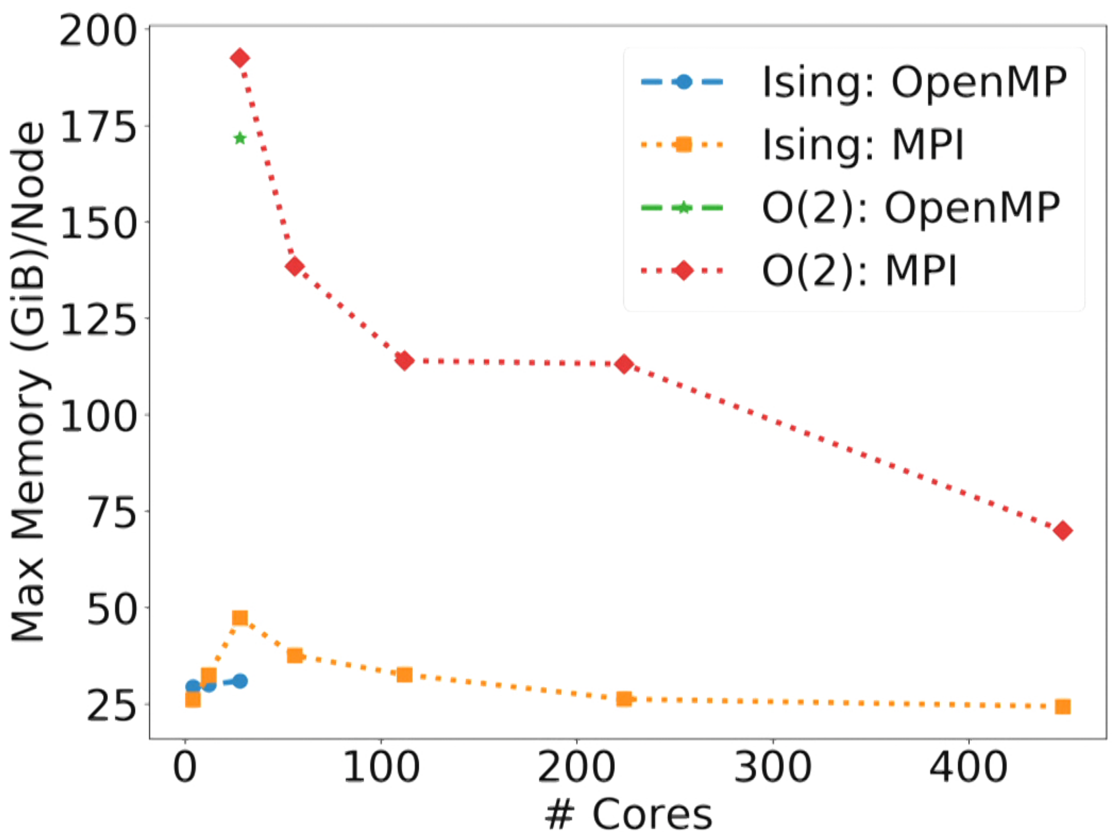
Q Caveats

- It is not as fast for large core counts.
  - Factor of 2-3 for O(2), n_max=18 with 448 cores at Yale
- However, you would only use large core counts for large problems.
- Previously, you would have a hard time fitting your large problem on the machine at all.
  - O(2), n_max=22 did not fit on Comet

Scaling on Large Machines



Memory Use



Max Memory (GiB)/Node
200
175
150
125
100
75
50
25
0
100
200
300
400
Cores
Ising: OpenMP
Ising: MPI
O(2): OpenMP
O(2): MPI

O(2) Remarks

- The O(2) project has been an excellent driver of progress for SDPB.
- It generated large, concrete benchmarks that people definitely wanted to solve.
- It highlighted bottlenecks when performing a complete bootstrap calculation, motivating improvements to block generation (scalar_blocks) and conversion from Mathematica SDP's to input (sdp2input)

Scalar Blocks

- Replaces Mathematica block generation
- Written in C++
- 111 times faster on 28 cores
- Available in the docker and images.

https://gitlab.com/bootstrapcollaboration/scalar_blocks

Spectrum Extraction

- Python script to extract the spectrum from the SDPB output
- Updated for new output format
- Clarified dependencies and made to work with python 2 or 3
- Also available in the docker and S images.

https://gitlab.com/bootstrapcollaboration/spectrum-extraction

XSEDE Proposal

- XSEDE is an NSF funded clearinghouse for supercomputer time at different centers.
- We wrote a proposal for the O(2) project for 5 million hours on the SDSC Comet cluster.

- Awarded 3 million hours
- Received 1.2 million hours
- Used up 200,000 hours in a few days

Cannon Cluster Proposal

- Harvard is standing up Cannon, a new cluster with 30,000 cores.
- They are looking for users who can thoroughly exercise the machine.
  - Science would be nice, but is not the driver
- Request for Proposals: Up to 3 days of compute time on the whole cluster.
- We submitted a proposal for ~1 million hours for more O(2) work.

Better Scaling

- The work so far has pushed the scalability of SDPB from ~20 cores to ~300.
- We have run jobs up to 768 cores.
- The rule of thumb is that each improvement by a factor of 10 takes significant effort.
- The next step will require careful benchmarking on large machines.

Why Such High Precision?

- I will be looking at a small stress tensor example. It seems non-trivial enough to be useful.
- You might expect to need only to resolve
  - The error threshold: 10^{-40}
  - The duality gap between the primal and dual solutions: 10^{-80}
- In practice, we need much, much higher precision.

What Breaks?

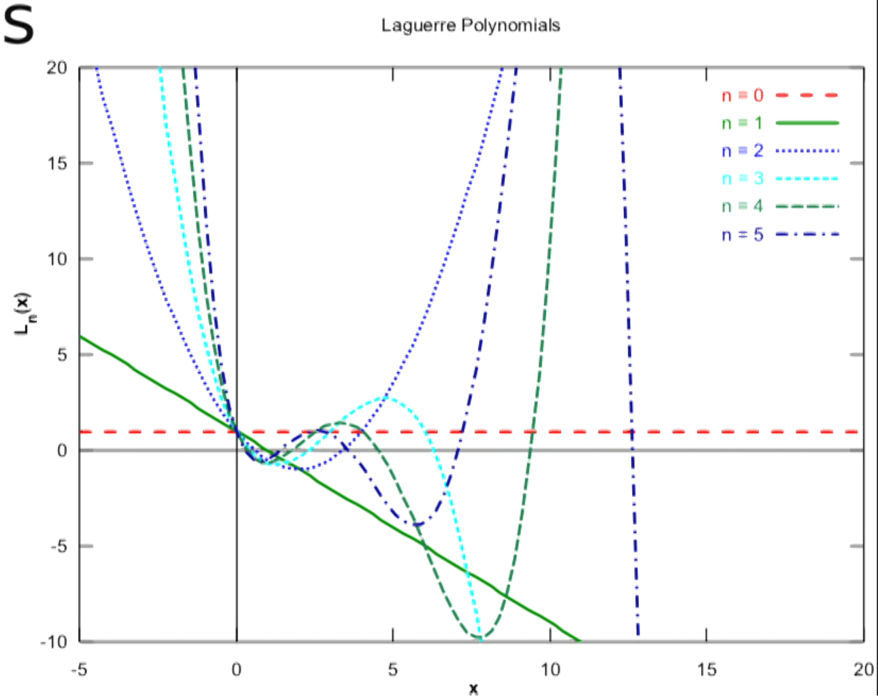
- The first thing that breaks when reducing precision is when solving
$$\begin{pmatrix} S & -B \\ B^T & 0 \end{pmatrix} \begin{pmatrix} dx \\ dy \end{pmatrix} = \begin{pmatrix} r_x \\ r_y \end{pmatrix}$$
- S has a block structure made up of symmetric positive-definite matrices.
- We use a Schur complement method, which involves inverting S first.

S is Ill-Conditioned

- When precision is low, S is no longer numerically positive.
- This is because S has a very bad condition number: 10^{180}
- This happens immediately, well before we do any real calculations.

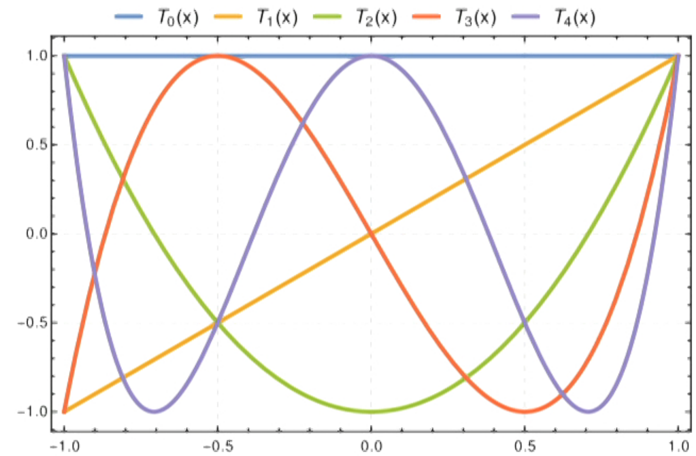
Bad Basis

- By default, we evaluate functions at the roots of Laguerre polynomials.
- Laguerre polynomials mimic exponentials, but the function we are approximating is well behaved over the domain.



Chebyshev Polynomials

- Chebyshev polynomials are ***very*** well behaved in their domain.
- We tried mapping the Chebyshev roots to the same interval.
- Evaluating functions at these new points yields a dramatic improvement in the condition number of S: 10^4

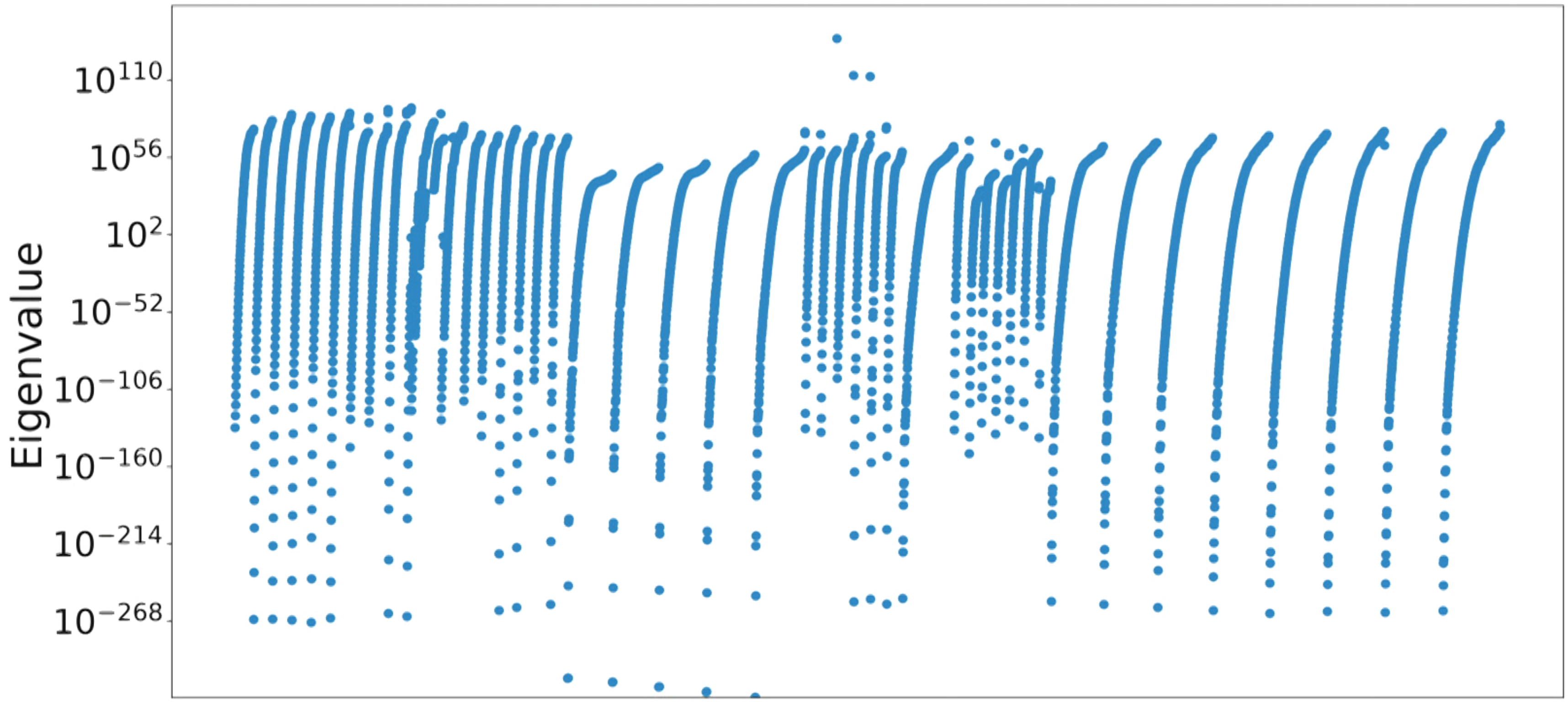


Not the Solution

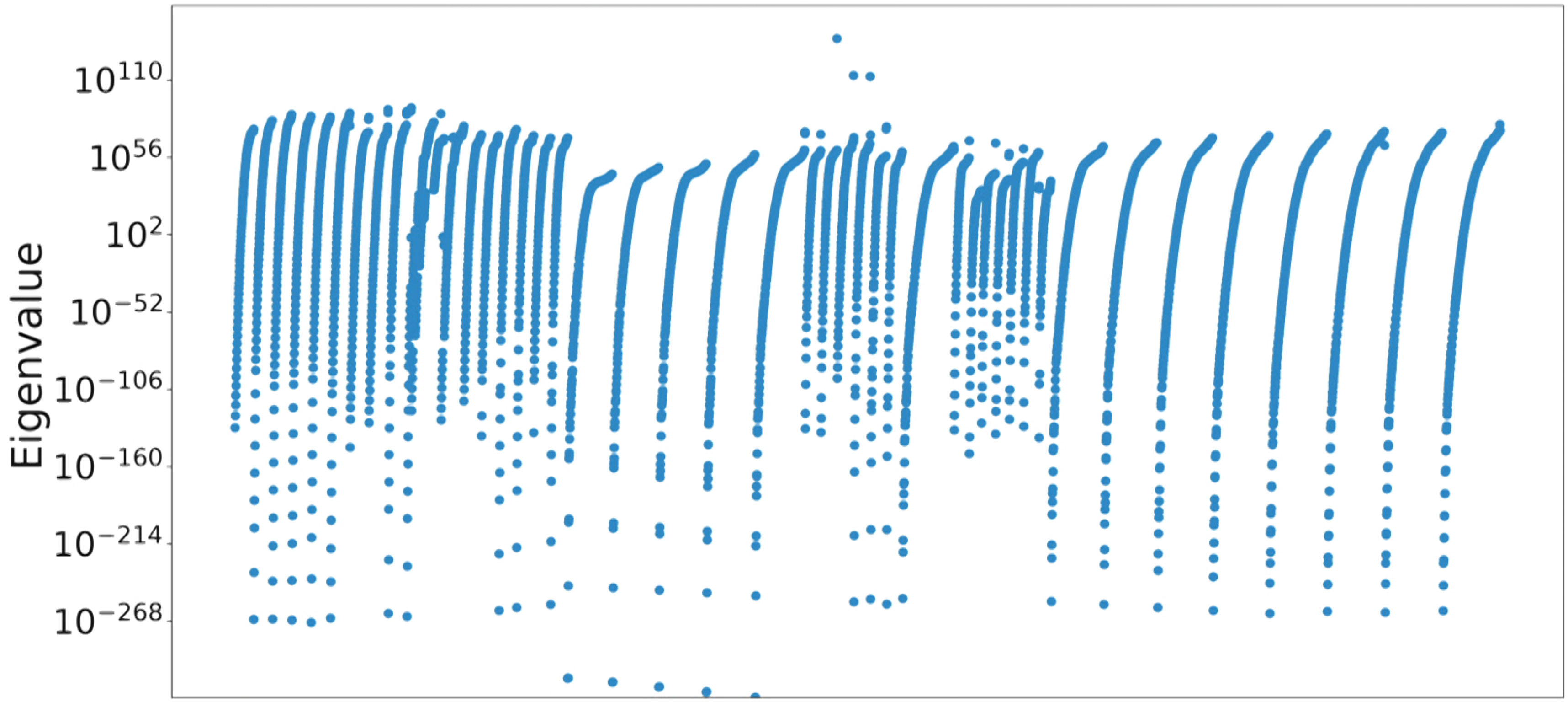
- Unfortunately, as SDPB converges on a solution, S again becomes very, very ill-conditioned:

$$10^{400}$$

Eigenvalue Spectrum of S



Eigenvalue Spectrum of S



Eigenvector Decomposition

- If we decompose dx and B into eigenvectors of S, it turns out that:

$$dx_i \propto {\lambda_i}^{-1/2} \qquad B_i \propto {\lambda_i}^{1/3}$$

- This implies $r_y = B \cdot dx \propto {\lambda_i}^{-1/6}$
- But r_y lives in a different space and at this point in the calculation, after a dual jump, is essentially zero.

Small Differences - Big Problems

- The scaling eventually breaks down at small λ_i, but there are still many cancellations over a large range of λ_i.
- It does mean that we can not just ignore small eigenvalues.
- So there is still more to understand.

New Work

- Gateways
- Cloud
- Job Management

Gateway

- A web interface to SDPB
  - Simple pointy-clicky
  - Scriptable (https POST)
- You upload input files.
- The Gateway submits these files to a supercomputer.
- You check progress from time to time.
- When finished, you collect the results

THE APACHE SOFTWARE FOUNDATION

Gateway Users

- This removes the need to understand supercomputers in order to do large calculations.
- Even for those who do understand them, there is no need to get any complex control software running there.
  - For example, Mathematica is not available on XSEDE

Gateway Implementation

- XSEDE is very interested in giving away free time for gateways.
  - I have implemented a gateway with them in the past (seismology simulations)
  - I have also run high traffic web sites with scientific users.
- We would essentially become a mini-funding agency, giving out SDPB time.
- Requires a committee to vet applications

Scaling to the

- Many different, large providers
- You can make use of enormous compute power.
  - Western Digital burned 8 million core-hours in 8 hours on hard drive simulations.
- Caltech seriously considered using the cloud instead of building their own supercomputer.

Cloud Details: $$$

- It costs serious money, but sometimes they give away time for free
- Compute is cheap-ish: 2-3 cents/core-hour
  - The Western Digital runs cost $137,307
- Storage is not cheap: ~2 cents/GB-month
- Transferring data into the cloud is free.
  - Getting data back out is not: ~9 cents/GB
- Good match for SDPB
  - long calculations and small outputs

Cloud Implementation

- Smaller, 1-node jobs may already work with Docker?
- Larger jobs require more thorough investigation and performance testing.
  - AWS ParallelCluster
    - No Infiniband. Maybe EFA is not terrible?
  - Azure Batch
    - Requires Intel compiler?
  - Not as big a market, so fewer choices

Managing Computations

- Assuming that all of these resources are available, we still need a way to manage all of the separate computations that combine into a single result.
- Existing approaches use a variation of Mathematica scripts or Haskell programs.
- We should figure out what is common to all of these approaches and automate that.

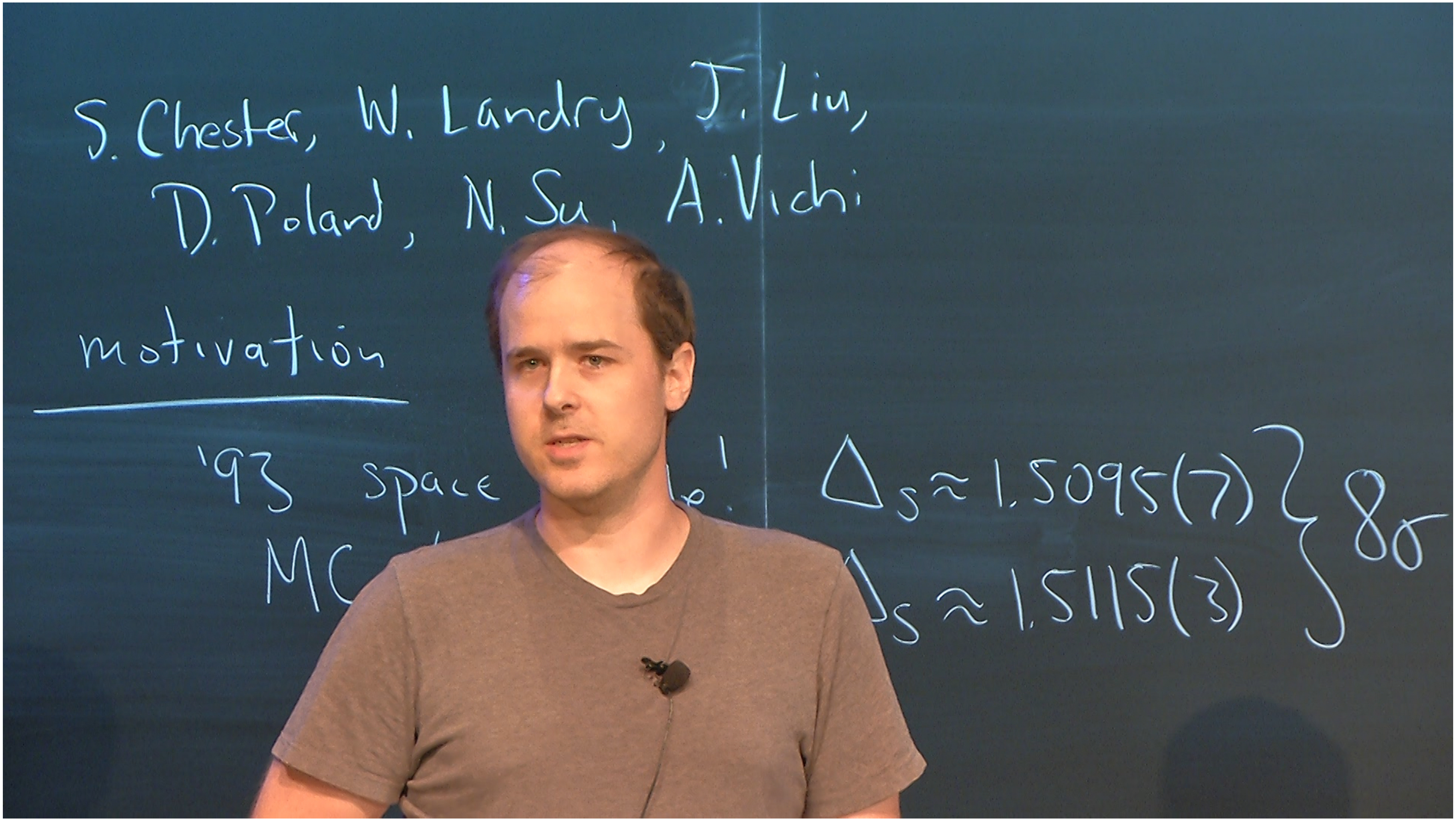
S. Chester, W. Landry, J. Liu,
D. Poland, N. Su, A. Vichi

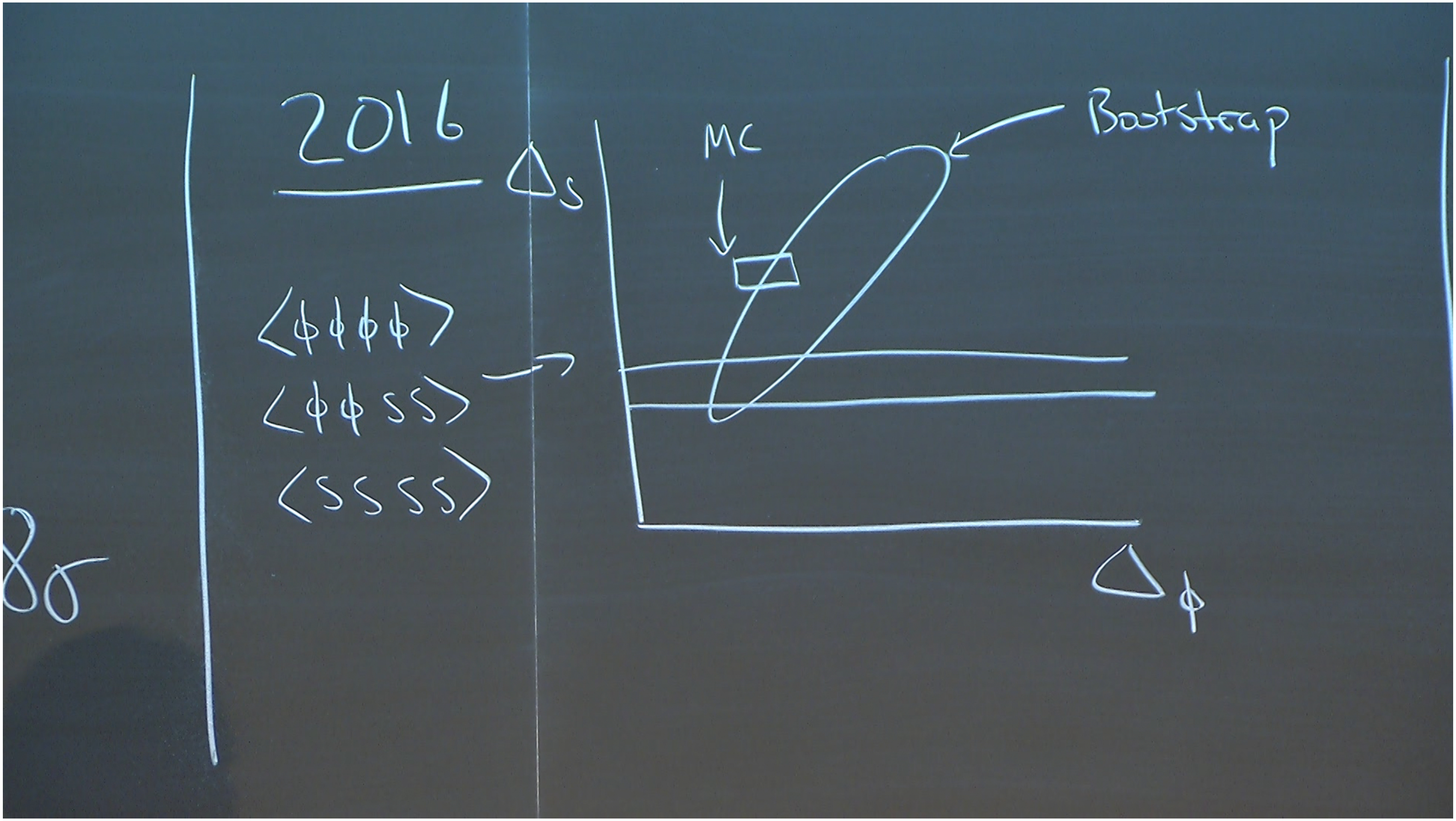
motivation

'93 space shuttle!

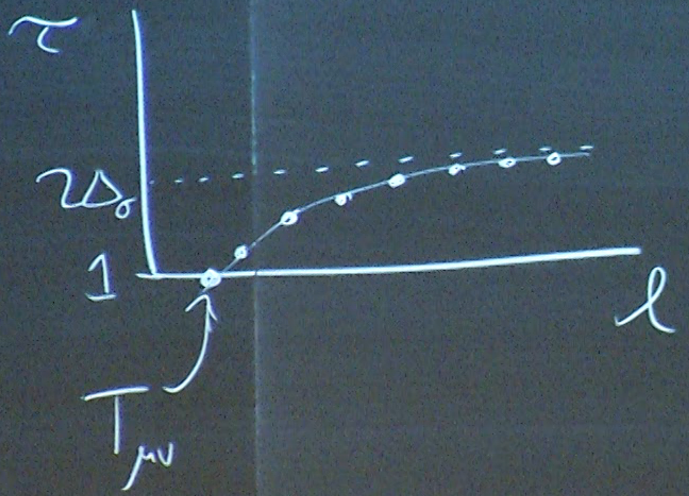
MC / high-temp

$$\left.\begin{array}{l}\Delta_s \approx 1.5095(7) \\ \Delta_s \approx 1.5115(3)\end{array}\right\} 8\sigma$$


S. Chester, W. Landry, J. Liu,
D. Poland, N. Su, A. Vichi
motivation
'93 space
MC
Δs ≈ 1.5095(7)
Δs ≈ 1.5115(3)
8σ


2016
$\langle \phi\phi\phi\phi \rangle$
$\langle \phi\phi s s \rangle$
$\langle s s s s \rangle$
Δ_s
MC
Bootstrap
Δ_ϕ

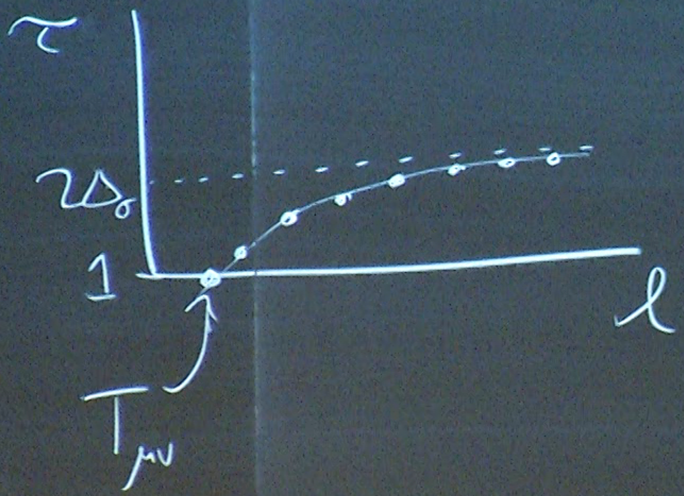
Lessons from Ising



$[\sigma\sigma]_0$

$$\Delta_{[\sigma\sigma]_0} = 2\Delta_\sigma - \frac{\Delta_\sigma^2}{C_T \ell} - \frac{f_{\sigma\sigma\epsilon}^2}{\ell^{\Delta_\epsilon}} + \cdots$$

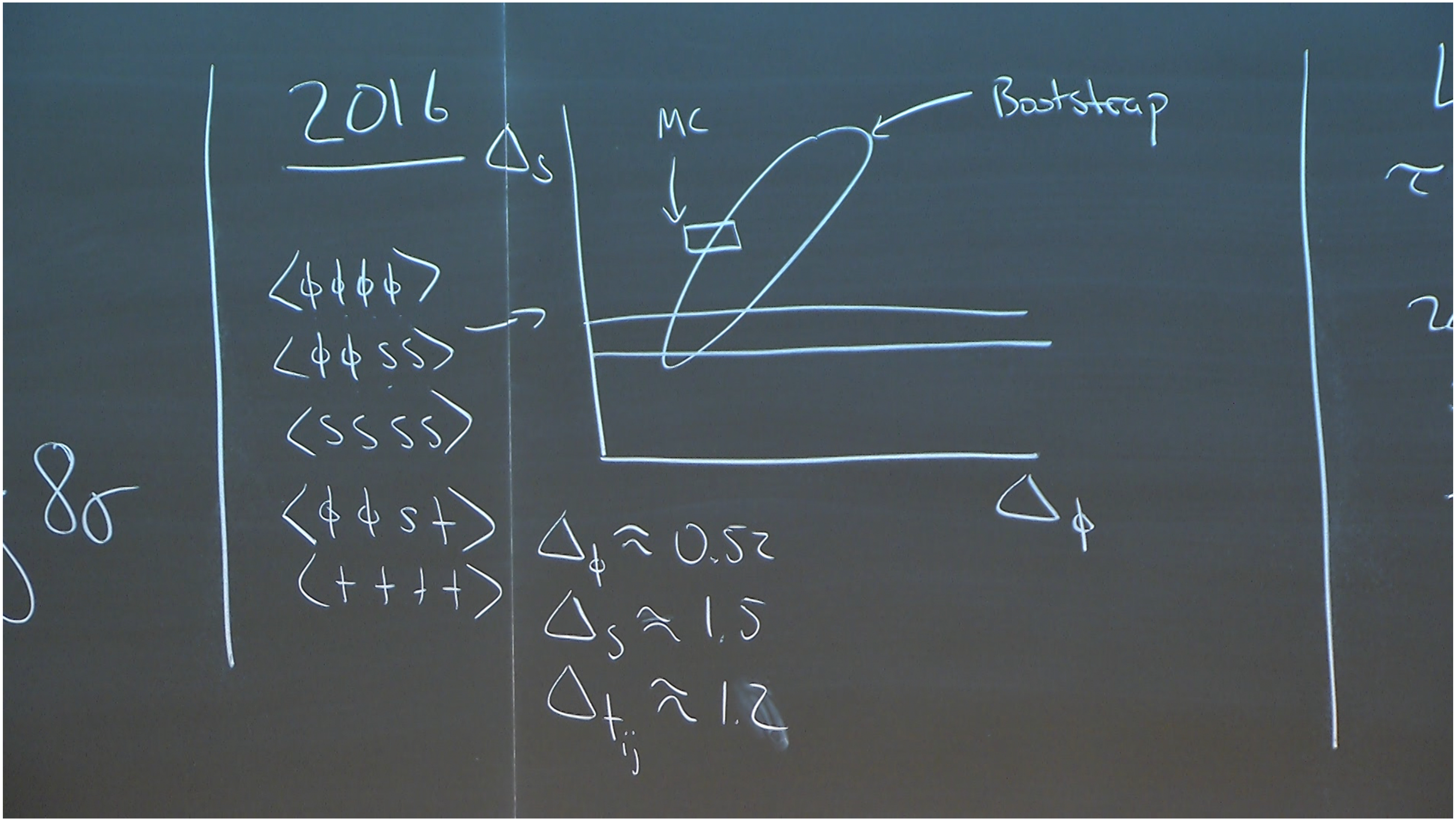
Lessons from Ising

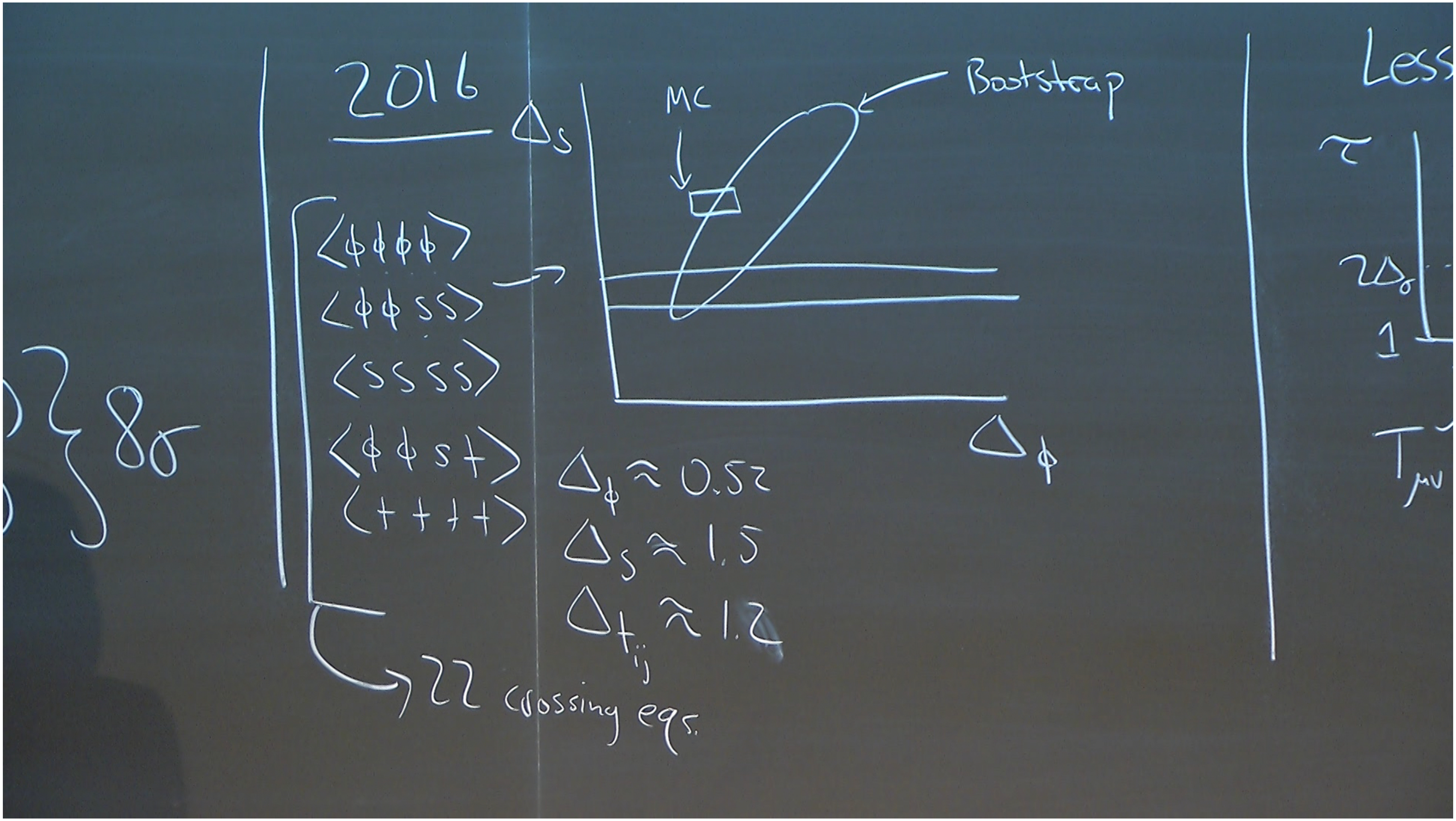


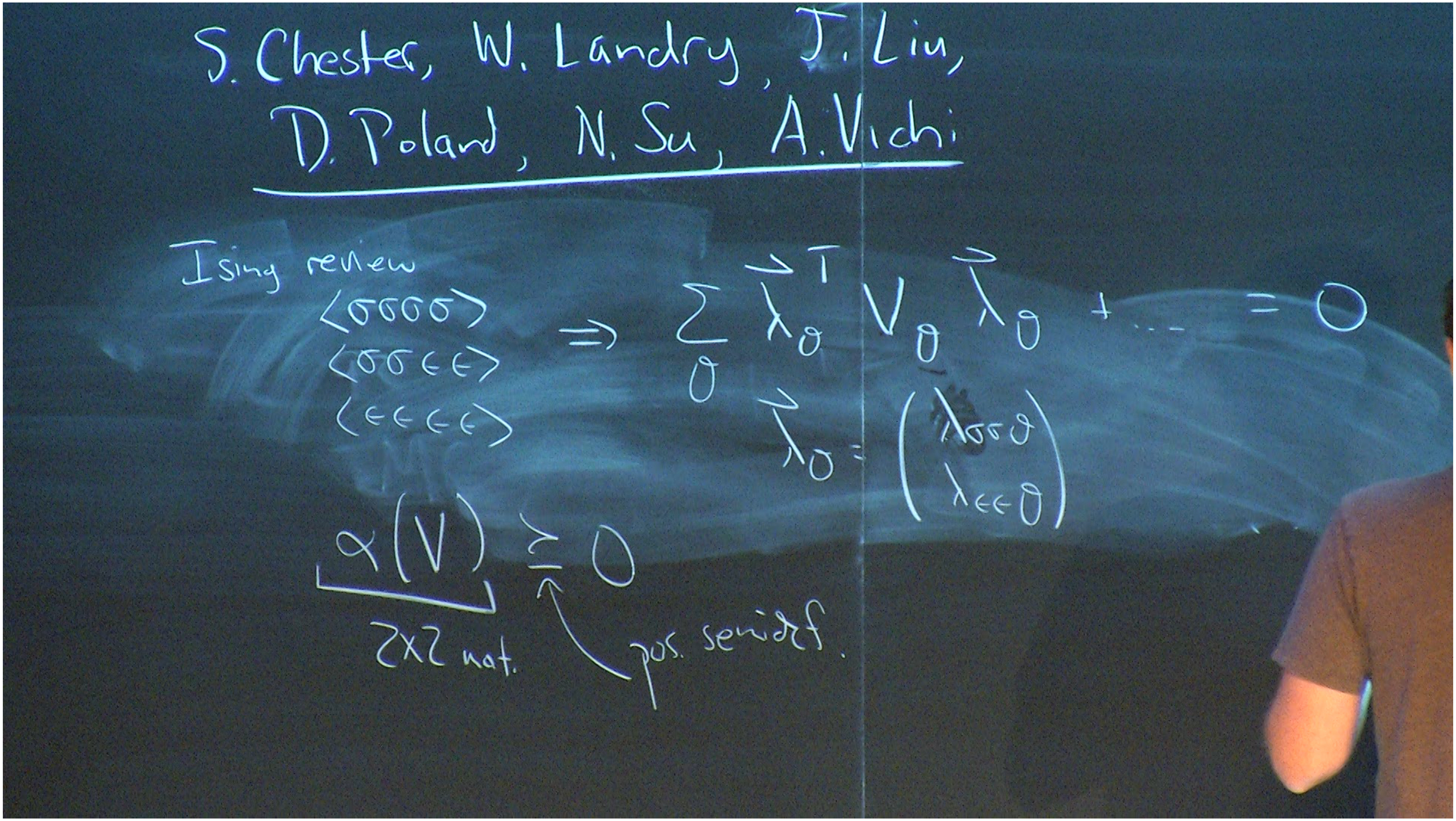
$[\sigma\sigma]_0$

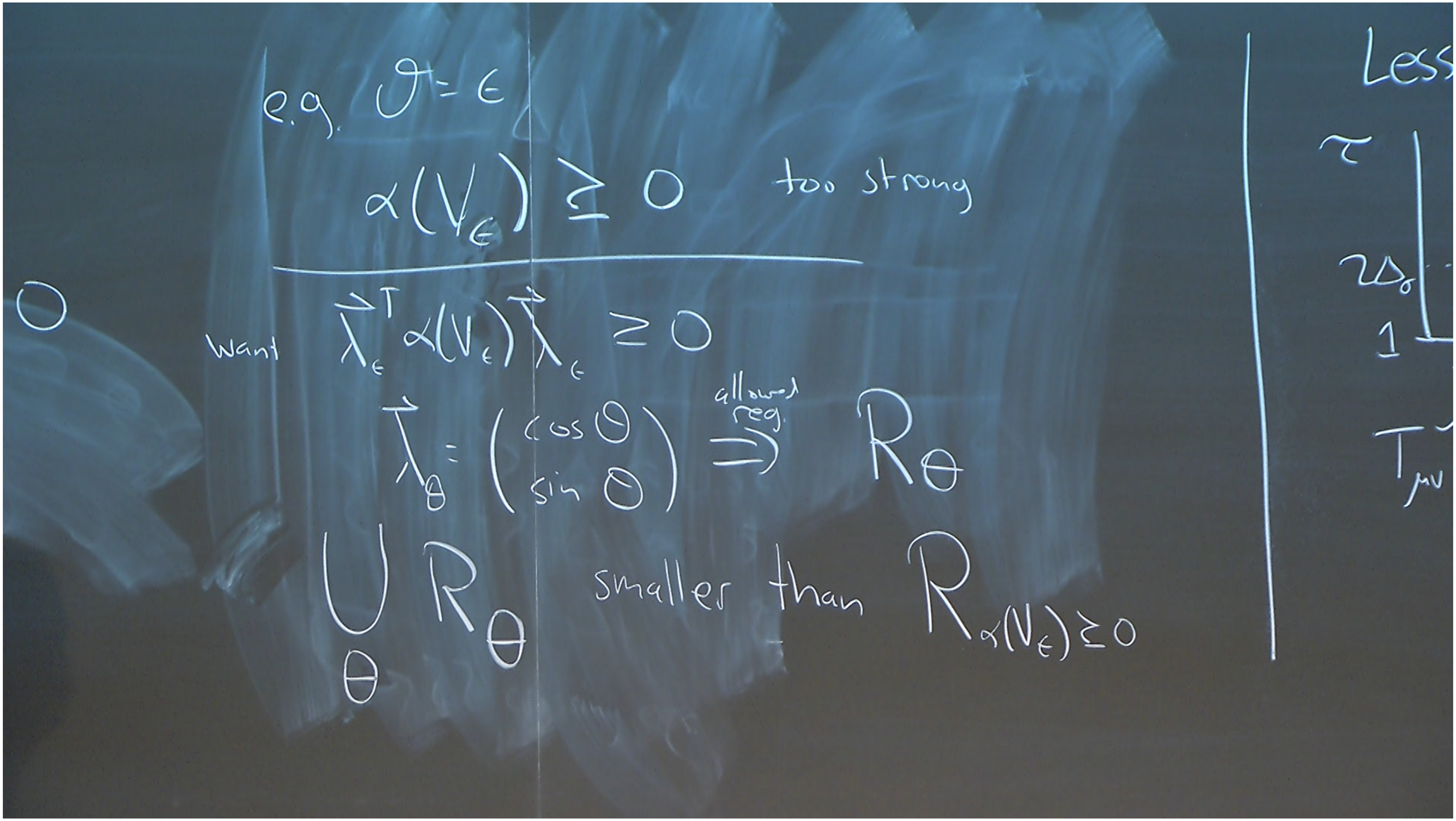
$$\Delta_{[\sigma\sigma]_0} = 2\Delta_\sigma - \frac{\Delta_\sigma^2}{c_T \ell} - \frac{f_{\sigma\sigma\epsilon}^2}{\ell^{\Delta_\epsilon}} + \cdots$$

$\ell = 2 \Rightarrow$ eq. $\Delta_\sigma, c_T, f_{\sigma\sigma\epsilon}, \Delta_\epsilon$


2016
⟨φφφφ⟩
⟨φφss⟩
⟨ssss⟩
⟨φφst⟩
⟨tttt⟩
Δ_s
MC
Bootstrap
Δ_ϕ
$\Delta_\phi \approx 0.52$
$\Delta_s \approx 1.5$
$\Delta_{t_{ij}} \approx 1.2$
8σ


2016
Δ_s
MC
Bootstrap
$\langle \phi\phi\phi\phi \rangle$
$\langle \phi\phi ss \rangle$
$\langle ssss \rangle$
$\langle \phi\phi st \rangle$
$\langle tttt \rangle$
22 crossing eqs.
Δ_ϕ
$\Delta_\phi \approx 0.52$
$\Delta_s \approx 1.5$
$\Delta_{t_{ij}} \approx 1.2$
8σ
Less
τ
$2\Delta_\sigma$
1
$T_{\mu\nu}$


S. Chester, W. Landry, J. Liu,
D. Poland, N. Su, A. Vichi
Ising review
⟨σσσσ⟩
⟨σσεε⟩
⟨εεεε⟩
⇒ ∑_O λ_O^T V_O λ_O + ... = 0
λ_O = (λ_σσO, λ_εεO)
α(V) ⪰ 0
2x2 mat.
pos. semidef.


e.g. $\vartheta = \epsilon$
$\alpha(V_\epsilon) \geq 0$ too strong
Want $\vec{\lambda}_\epsilon^T \alpha(V_\epsilon) \vec{\lambda}_\epsilon \geq 0$
$\vec{\lambda}_\theta = \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} \Rightarrow R_\theta$
allowed reg.
$\bigcup_\theta R_\theta$ smaller than $R_{\alpha(V_\epsilon) \geq 0}$
Less
τ
$2\Delta_\sigma$
1
$T_{\mu\nu}$

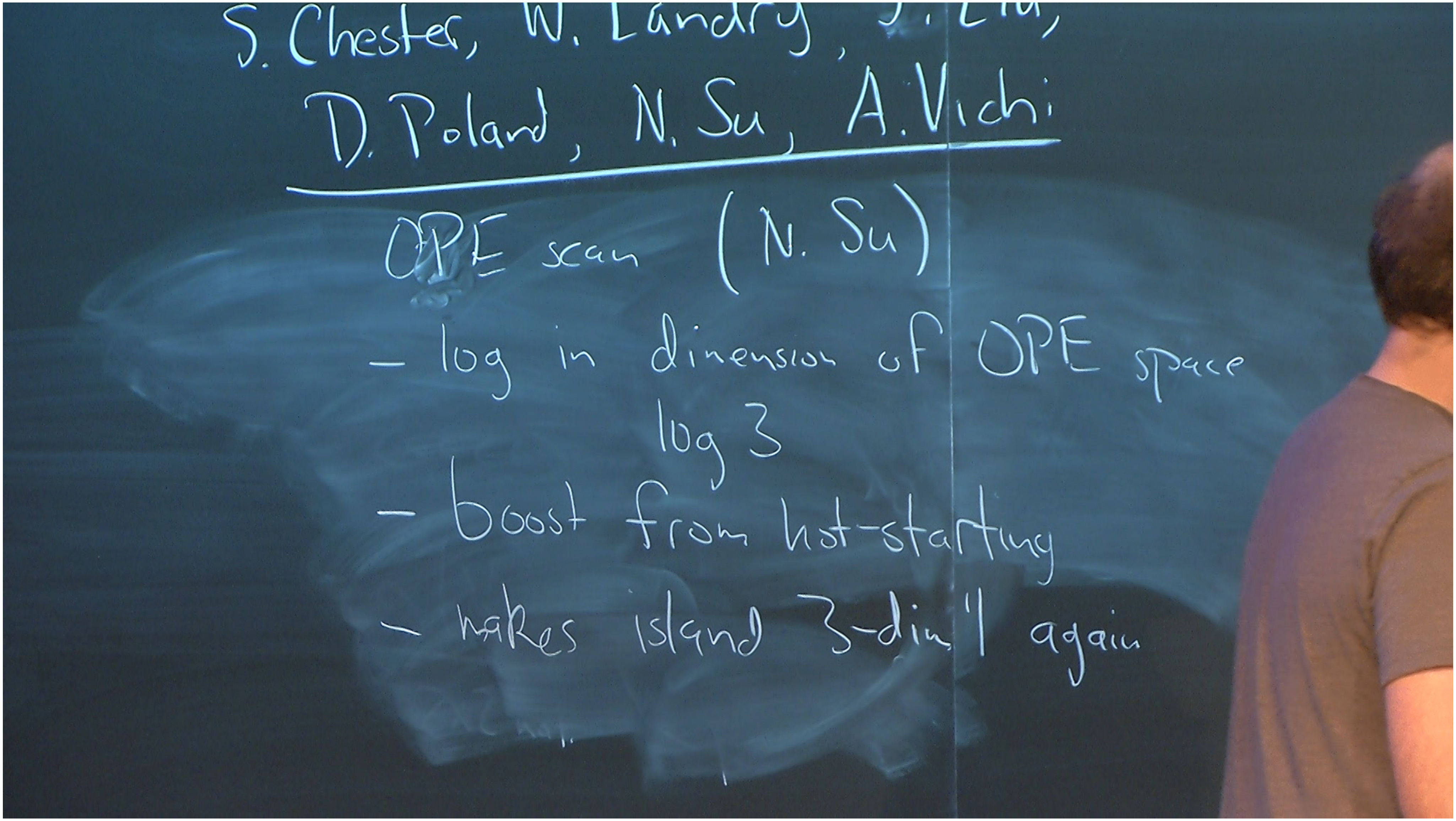
$$\vec{\lambda}_{ext} = \begin{pmatrix} \lambda_{sss} \\ \lambda_{\phi\phi s} \\ \lambda_{tts} \\ \lambda_{\phi t t} \end{pmatrix}$$

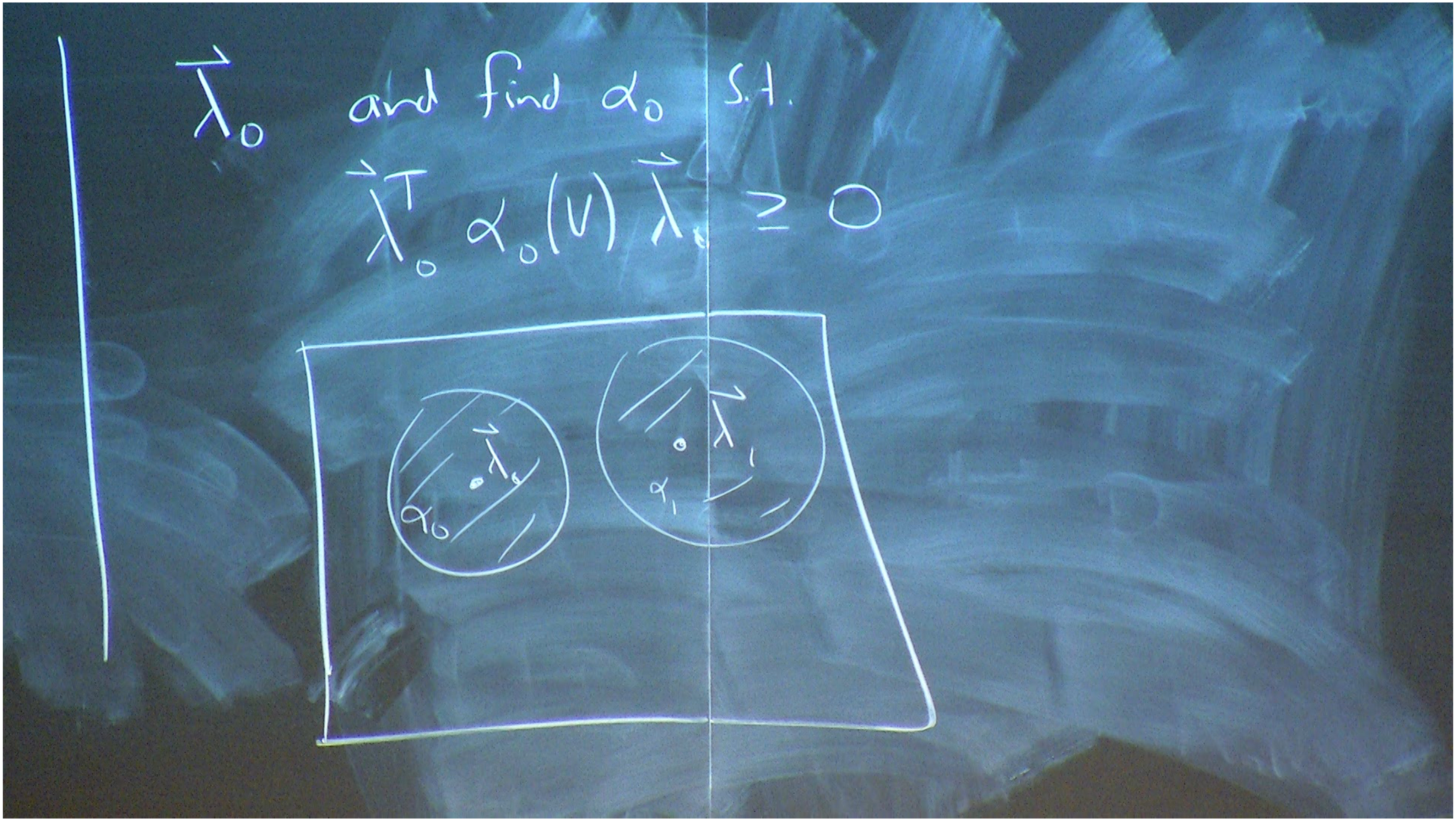
$$\vec{\lambda}_{ext}^{T} \, V_{ext} \, \vec{\lambda}_{ext} + \ldots = 0 \quad \text{(crossing)}$$

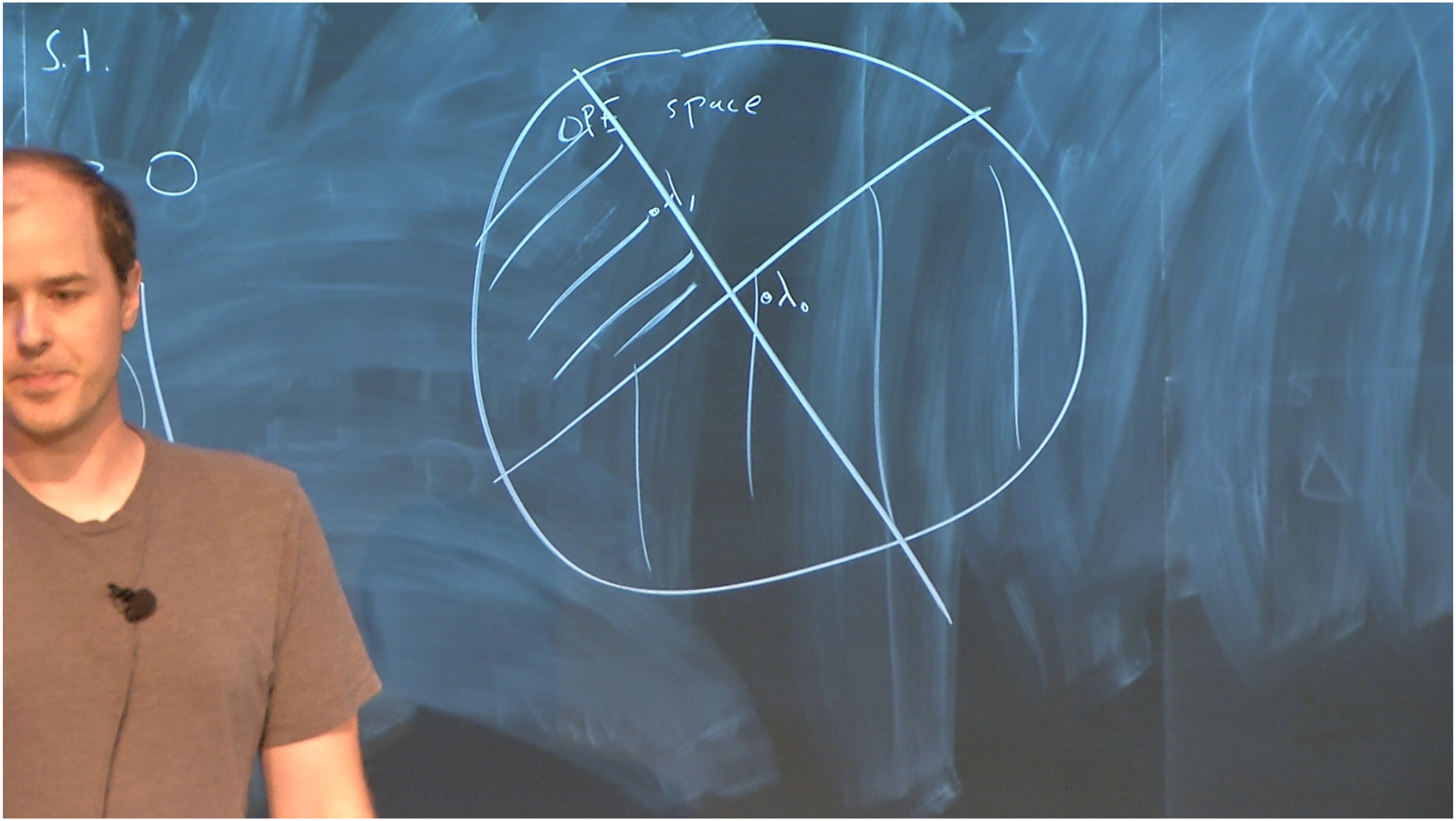
scan over direction of $\vec{\lambda}_{ext}$

island lives in 6-dimensions

$\Delta_\phi, \Delta_s, \Delta_t, \mathbb{RP}^3$


S. Chester, W. Landry,
D. Poland, N. Su, A. Vichi
OPE scan (N. Su)
– log in dimension of OPE space
log 3
– boost from hot-starting
– makes island 3-dim'l again


$\vec{\lambda}_0$ and find α_0 s.t.
$\vec{\lambda}_0^T \alpha_0(V) \vec{\lambda}_0 \geq 0$
α_0
$\vec{\lambda}_0$
$\vec{\lambda}_1$
α_1


S.t.
OPE space
λ_1
λ_0

When I tried to google this workshop

bootstrap 2019

Should i bother learning Bootstrap in 2019? : webdev - Reddit

https://www.reddit.com/r/webdev/.../should_i_bother_learning_bootstrap_in_2019/

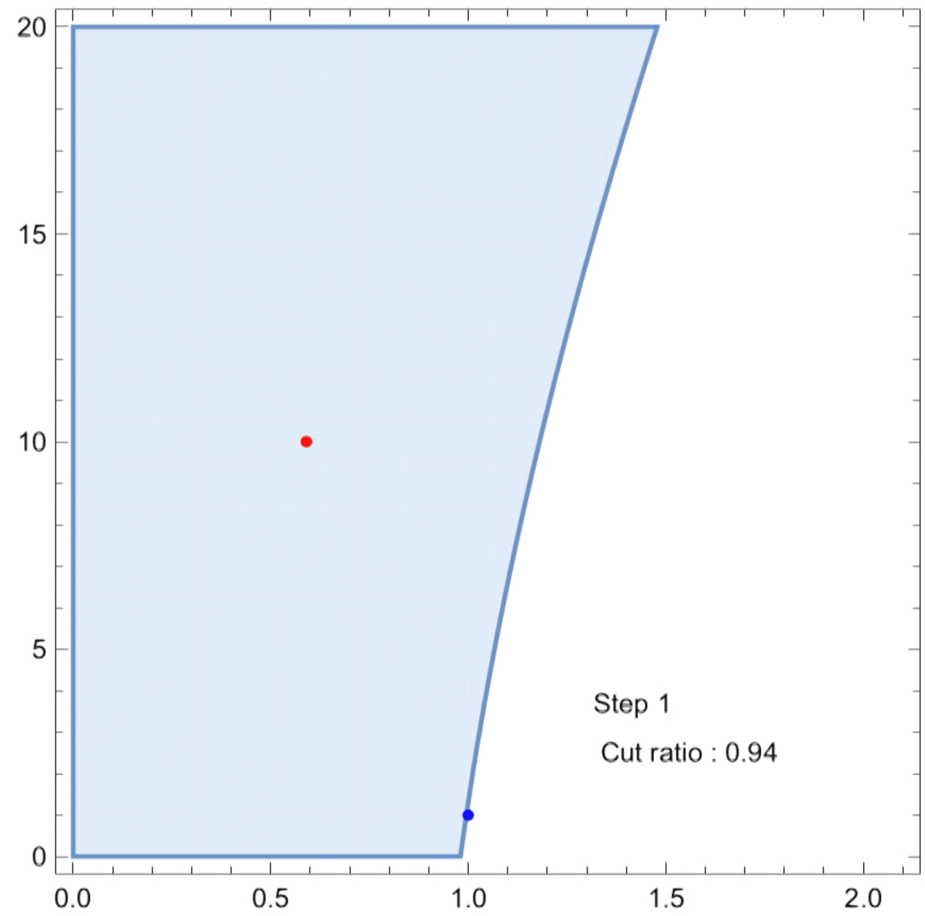
Jan 15, 2019 - 6 posts - 6 authors

I stopped using **bootstrap** a year ago. I don't like all of the css classes that it takes up. I created my own lightweight css framework that is very ...

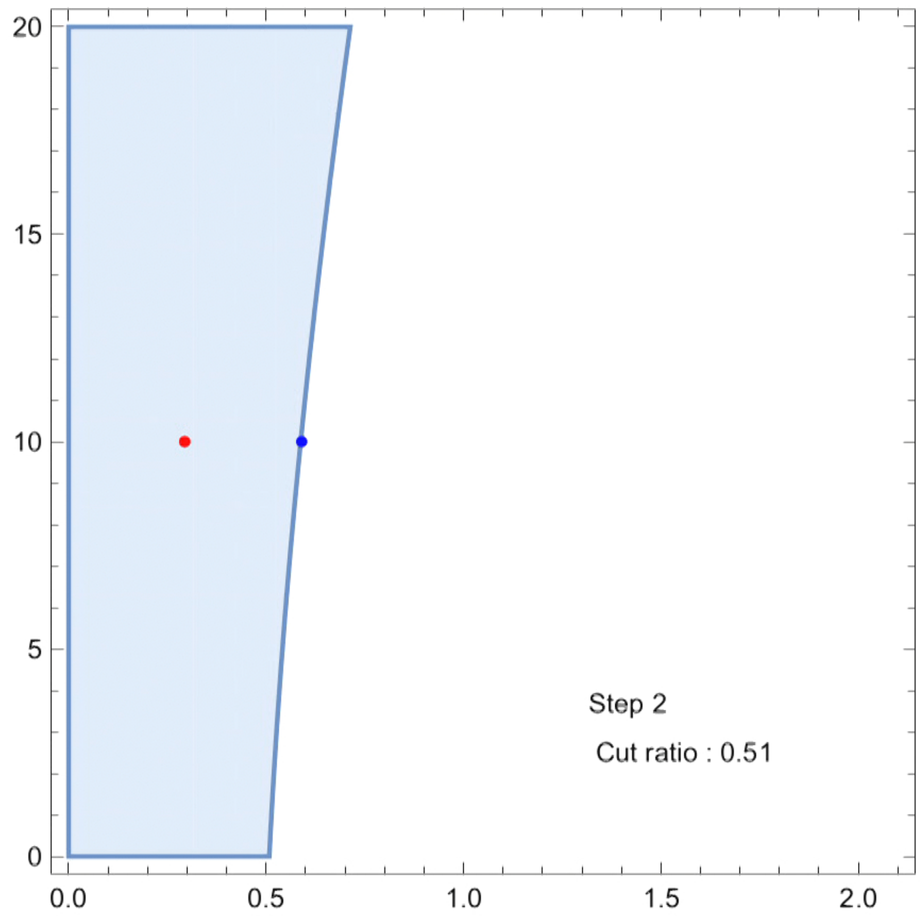
What's your opinion of **bootstrap**+jquery in **2019**? : javascript	Jan. 24, 2019
Is **Bootstrap** still important to learn? : webdev	Apr. 29, 2018
Angular Material Vs **Bootstrap** 4 for new project : Frontend	Mar. 26, 2019

More results from www.reddit.com

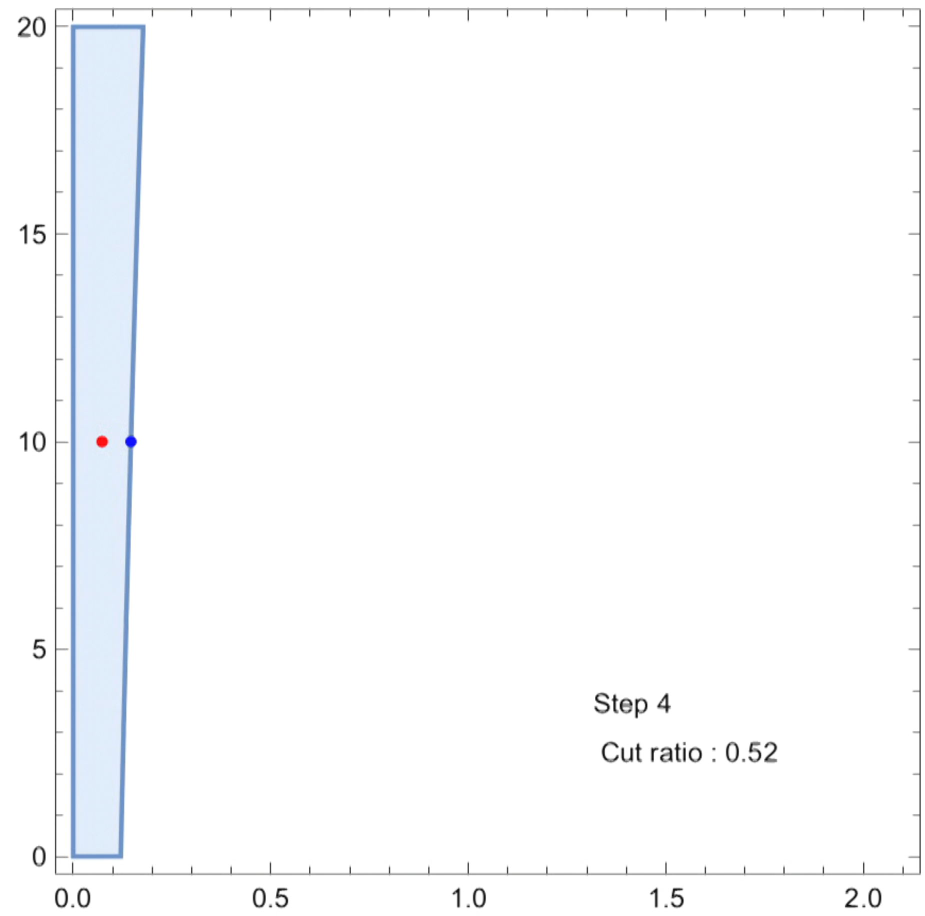
OPE search example figure: Ning



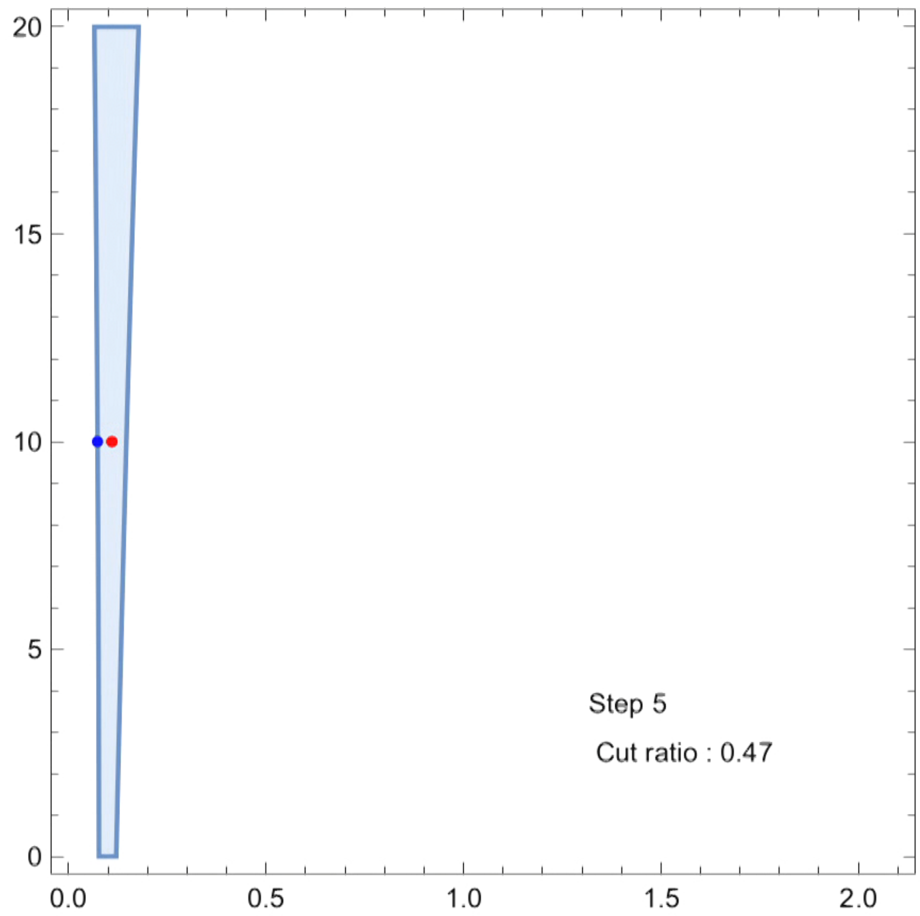
OPE search example figure: Ning



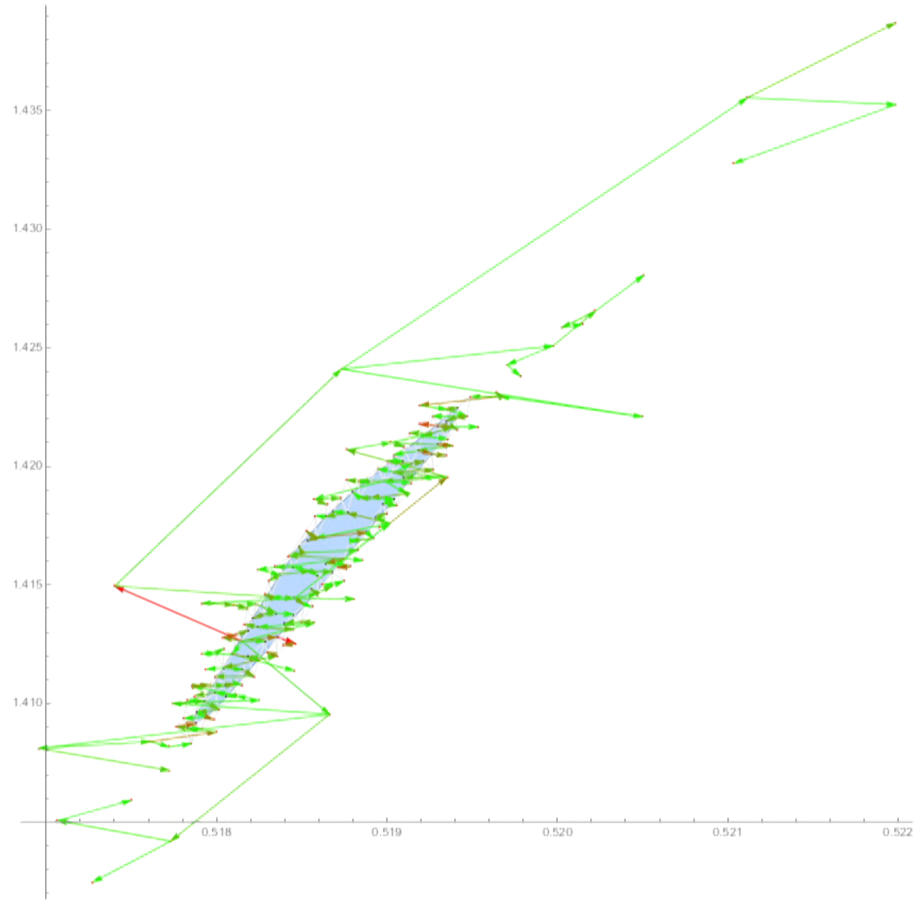
OPE search example figure: Ning



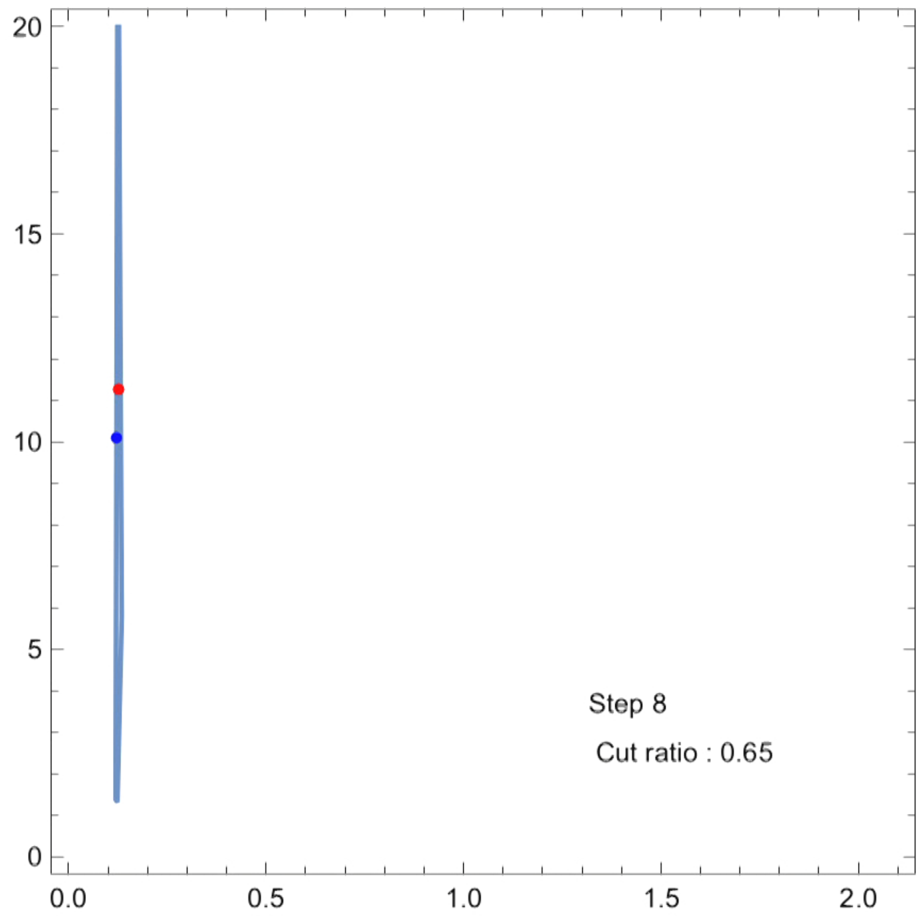
OPE search example figure: Ning



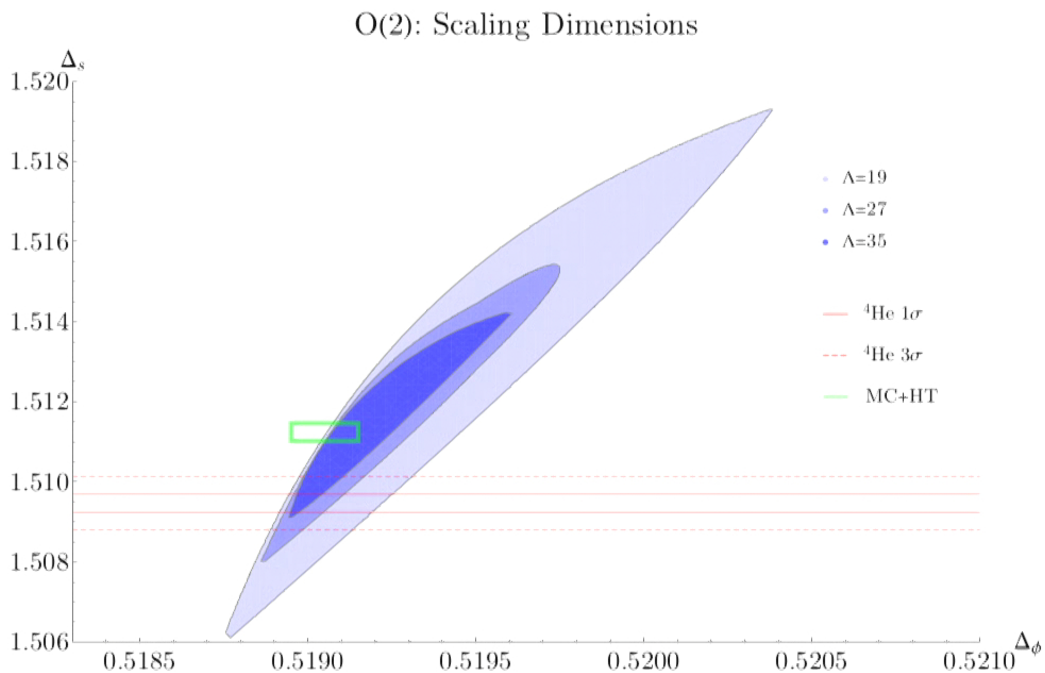
Hot-starting effectiveness figure: Ning



OPE search example figure: Ning

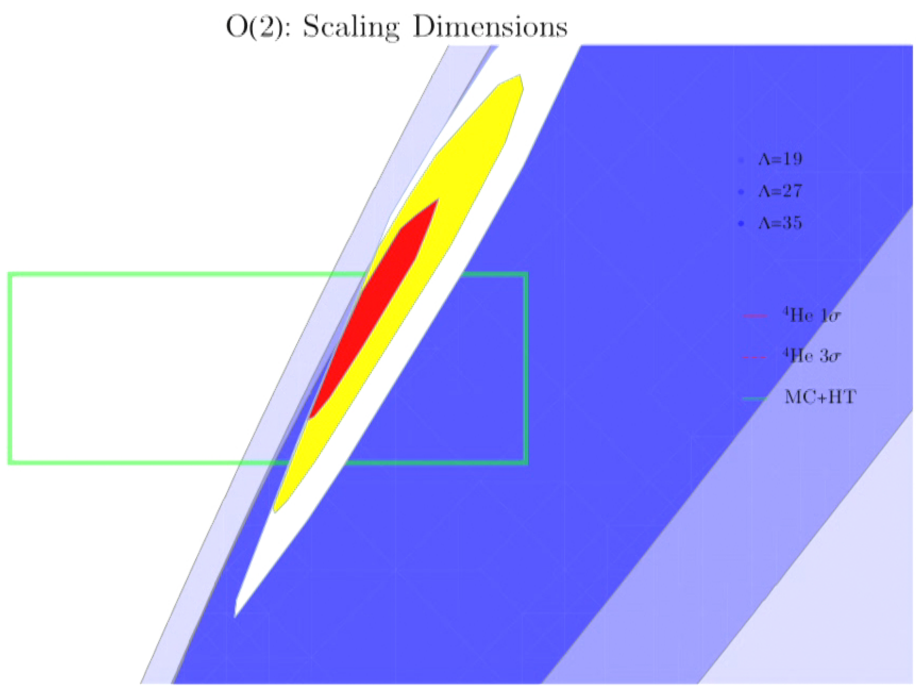


Old $O(2)$ results Kos, Poland, DSD, Vichi '16



Four-point functions of s, ϕ_i. 7 crossing equations with up to 2×2 matrices. Functionals with size up to 1197.

New results (preliminary) Chester, Landry, Liu, Poland, DSD, Su, Vichi '19



Nature is wrong! (jk — our results appear consistent with experimental data).