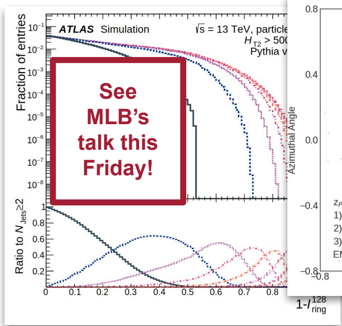
SPECTER: Efficient Evaluation of the Spectral EMD

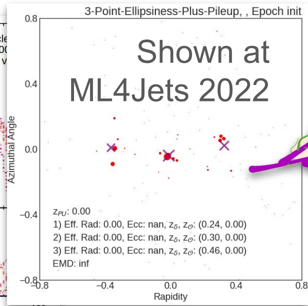
Rikab Gambhir

Email me questions at rikab@mit.edu!
Based on [**RG**, Larkoski, Thaler, 23XX.XXXX]

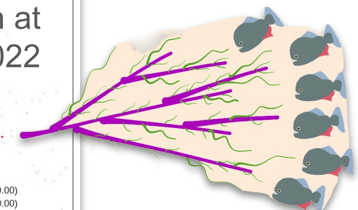
The **Wasserstein Metric,** a.ka. **Earth/Energy Mover's Distance (EMD)** has seen increasing interest in jet physics:



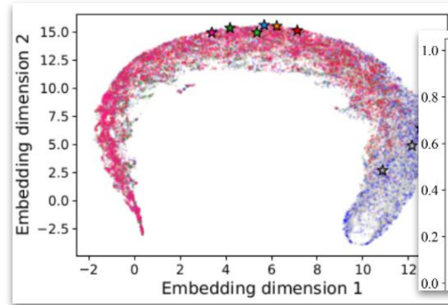
[ATLAS, 2305.16930]



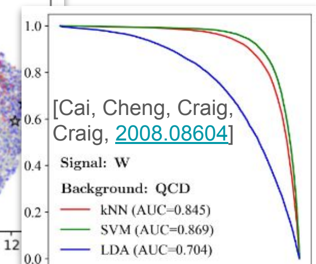
[Ba, Dogra, **RG**, Tasissa, Thaler, 2302.12266]

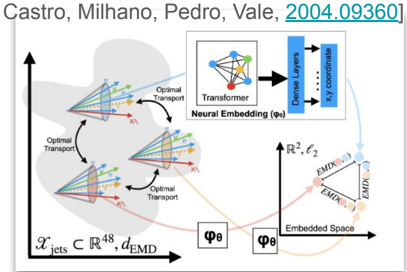


[Alipour-Fard, Komiske, Metodiev, Thaler, 2305.00989]

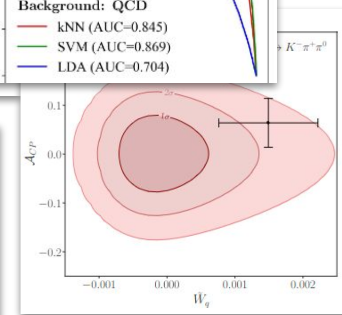


[Romao, Castro, Milhano, Pedro, Vale, 2004.09360]

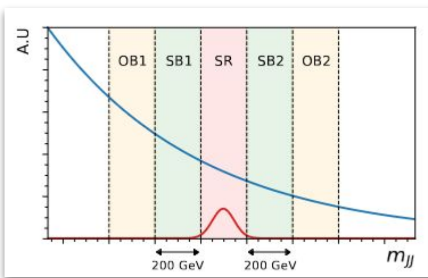




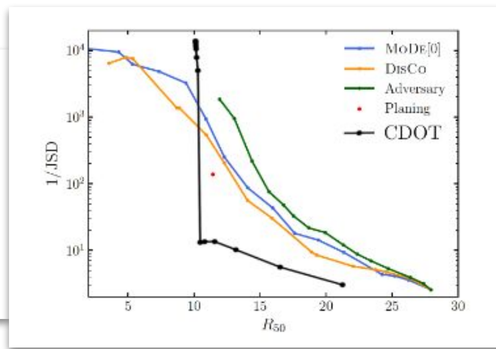
[Park, Harris, Ostdiek, 2208.05484]



[Davis, Menzo, Youssef, Zupan, 2301.13211]



[Raine, Klein, Sengupta, Golling, 2203.09470]



[Chakravarti, Kuusela, Wasserman, ML4Jets 2022]

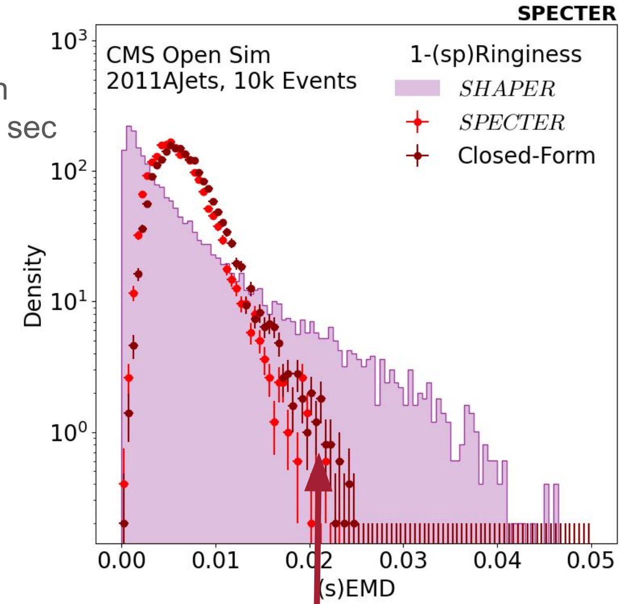
both computationally, and also in QCD calculations…

But! The EMD is **hard/expensive to calculate**, and even **harder to minimize**…

Not an exhaustive list, let me know if I haven't included your recent EMD application!

Today …

Old Method: ~ 3 hours
New (Numeric): ~15 min
New (Closed Form): ~3 sec
On my laptop's CPU



Making the EMD and associated observables *easier* and *faster* to calculate using the **Spectral EMD** (SEMD) and ***SPECTER***

With these tools, we can calculate this dark curve in *seconds*, equivalent to ~10^6 OT problems*!

$$\mathrm{SEMD}_{\beta,p=2}(s_A, s_B) = \sum_{i<j\in\mathcal{E}_A} 2E_iE_j\omega_{ij}^2 + \sum_{i<j\in\mathcal{E}_B} 2E_iE_j\omega_{ij}^2$$
$$-2\sum_{n\in\mathcal{E}_A^2,\, l\in\mathcal{E}_B^2} \omega_n\omega_l \left(\min\left[S_A(\omega_n^+), S_B(\omega_l^+)\right] - \max\left[S_A(\omega_n^-), S_B(\omega_l^-)\right]\right)$$
$$\times\,\Theta\left(S_A(\omega_n^+) - S_B(\omega_l^-)\right)\Theta\left(S_B(\omega_l^+) - S_A(\omega_n^-)\right),$$

SPECTER

Logo made with DALL-E. Preliminary.

*10^6 = 10k events × ~150 epochs

Central Idea: use the **Spectral EMD**[1]!

For $p = 2$, possible to find an exact solution for the optimal transport problem on the spectral representation of events:

$$\boxed{\begin{aligned}\mathrm{SEMD}_{\beta,p=2}(s_A,s_B) = &\sum_{i<j\in\mathcal{E}_A} 2E_iE_j\omega_{ij}^2 + \sum_{i<j\in\mathcal{E}_B} 2E_iE_j\omega_{ij}^2 \\ -2 \sum_{n\in\mathcal{E}_A^2,\, l\in\mathcal{E}_B^2} &\omega_n\omega_l \left(\min\left[S_A(\omega_n^+), S_B(\omega_l^+)\right] - \max\left[S_A(\omega_n^-), S_B(\omega_l^-)\right]\right) \\ &\times \Theta\left(S_A(\omega_n^+) - S_B(\omega_l^-)\right)\Theta\left(S_B(\omega_l^+) - S_A(\omega_n^-)\right) ,\end{aligned}}$$

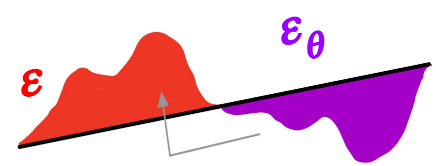
Can be computed exactly in $O(N^2 \log N)$, as opposed to the full EMD in $O(N^3)$
Closed form, easy derivatives and extremely easy to calculate programmatically!

Our framework for doing this, built in Python with JAX ← ***SPECTER***

See also: Sinkhorn, Sliced Wasserstein, WGANs, Linearized EMDs, …

[1][Larkoski, Thaler, 2305.03751]
[Larkoski, **RG**, Thaler, 23XX.XXXX]

Technical Details …



For p = 2, possible … problem on the sp…

For events A, B, the p **spectral EMD** is defined as (1D OT!):

EMD = Work done to move "dirt" optimally

$$\mathrm{SEMD}_{\beta,p}(s_A, s_B) \equiv \int_0^{E^2_{\mathrm{tot}}} dE^2 \left|S_A^{-1}(E^2) - S_B^{-1}(E^2)\right|^p$$

$$\mathrm{SEMD}_{\beta,p=2}(s_A, s_B) = \sum_{i<j\in\mathcal{E}_A} 2E_iE_j\omega_{ij}^2 + \sum_{i<j\in\mathcal{E}_B} 2E_iE_j\omega_{ij}^2$$
$$-2\sum_{n\in\mathcal{E}_A^2,\, l\in\mathcal{E}_B^2} \omega_n\omega_l\left(\min\left[S_A(\omega_n^+), S_B(\omega_l^+)\right] - \max\left[S_A(\omega_n^-), S_B(\omega_l^-)\right]\right)$$
$$\times\, \Theta\left(S_A(\omega_n^+) - S_B(\omega_l^-)\right)\Theta\left(S_B(\omega_l^+) - S_A(\omega_n^-)\right),$$

S = **cumulative spectral function**
± indicates whether or not to include ω in the sum

The trick: Sum over pairs n of particles within each event.

Looks like $O(N^4)$, but with clever sorting & indexing in 1D, reduces to ***$O(N^2)$***!

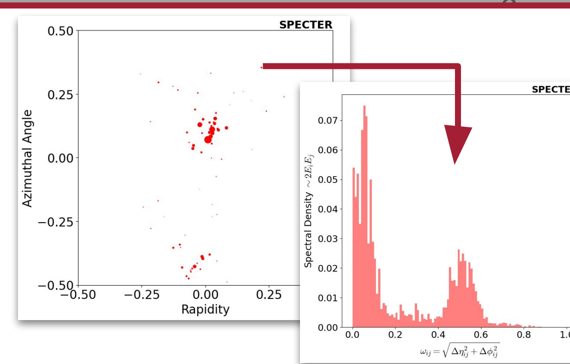
The **spectral density function**

$$s(\omega) = \sum_{i=1}^{N}\sum_{j=1}^{N} E_i\, E_j\, \delta\big(\omega - \omega(\hat{n}_i, \hat{n}_j)\big)$$

Pairwise Distances

Reduces events to 1D, while preserving all* information about the event, up to translations and rotations.

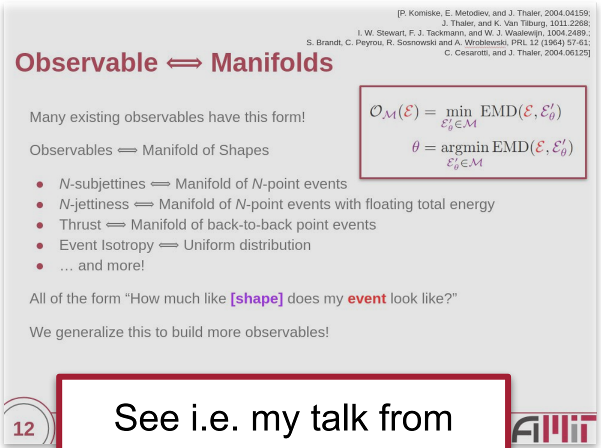
*up to measure 0, but important degeneracies, ask me about this later!



… EMDs, …

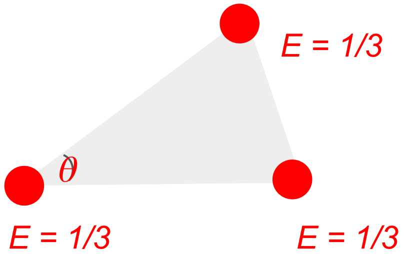
With a geometry based metric, we can now define IRC-safe **shape observables** by finding events that minimize the metric:

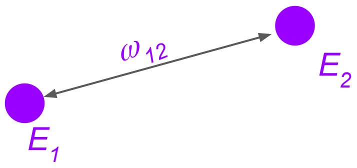
$$\boxed{\mathcal{O}(\mathcal{E}) = \min_{\mathcal{E}' \in \mathcal{M}} [\text{SEMD}(s(\mathcal{E}), s(\mathcal{E}')]}$$

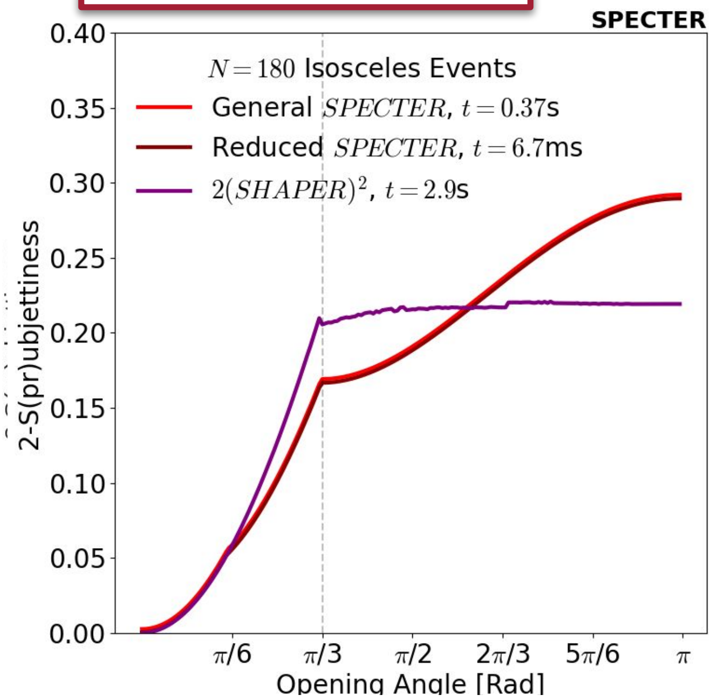


See i.e. my talk from ML4Jets 2022!

e.g. How 2-pointy are jets? (*2-subjettiness*)
Minimize the metric over 2-particle events





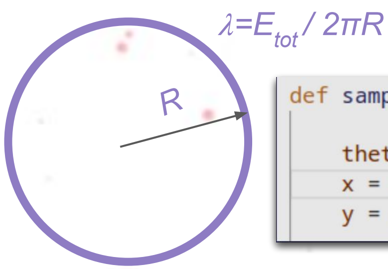


Pictured: Approximating the *2-subjettiness* with the spectral *2-s(pr)ubjettiness*, which is much faster!
Closed form: Only need to solve for $2E_1E_2$

Not all spectral functions correspond to a physical event, so we must choose whether we minimize over events or spectral functions – ask me about "ghost events" later!

Full Example: How "ring-like" are jets?

Step 1: Define the shape with parameters




```
def sample_circle(params, N, seed):

    thetas = jax.random.uniform(seed, shape
    x = params["Radius"] * jnp.cos(thetas)
    y = params["Radius"] * jnp.sin(thetas)
```

Unlike ordinary EMD, not necessary to specify center / orientation!

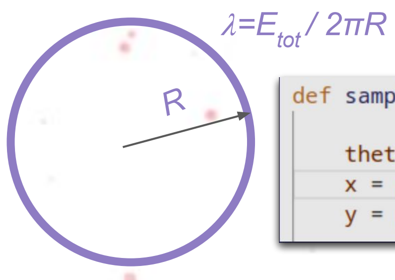
Shapes are parameterized distributions of energy on the detector space.

Many of your favorite observables, like *N*-(sub)jettiness, thrust, and angularities take the form of finding the shape that best fits an event's energy distribution.

Custom shapes define custom IRC-safe observables – to define a shape, all you need is to define a parameterized energy distribution and how to sample points from it!

Full Example: How "ring-like" are jets?

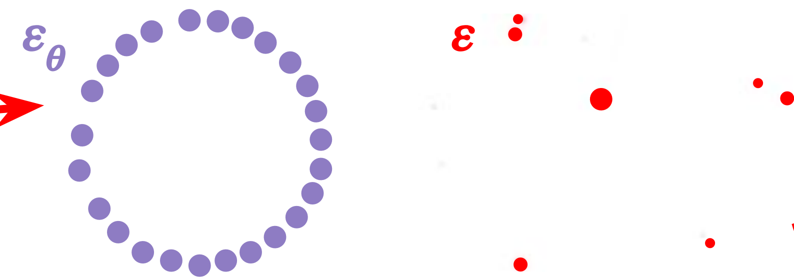
Step 1: Define the shape with parameters




```
def sample_circle(params, N, seed):

    thetas = jax.random.uniform(seed, shape
    x = params["Radius"] * jnp.cos(thetas)
    y = params["Radius"] * jnp.sin(thetas)
```

Unlike ordinary EMD, not necessary to specify center / orientation!

Step 2: Sample from Parameterized Shapes



Step 3: Calculate the spectral metric between events and shapes

$$\mathrm{SEMD}_{\beta,p=2}(s_A, s_B) = \sum_{i<j\in\mathcal{E}_A} 2E_iE_j\omega_{ij}^2 + \sum_{i<j\in\mathcal{E}_B} 2E_iE_j\omega_{ij}^2$$
$$- 2\sum_{n\in\mathcal{E}_A^2,\, l\in\mathcal{E}_B^2} \omega_n\omega_l \left(\min\left[S_A(\omega_n^+), S_B(\omega_l^+)\right] - \max\left[S_A(\omega_n^-), S_B(\omega_l^-)\right]\right)$$
$$\times\, \Theta\left(S_A(\omega_n^+) - S_B(\omega_l^-)\right)\Theta\left(S_B(\omega_l^+) - S_A(\omega_n^-)\right),$$

Key difference from previous work: We use the SEMD, *not* the EMD!

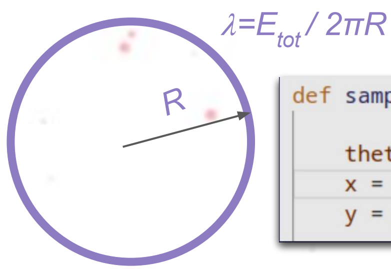
The p = 2 spectral EMD between two sets of discrete points has a closed-form solution with only binary discrete minimizations.

We discretize our shape by randomly sampling points from it.

If the spectral functions are sorted, can compute the SEMD in ~$O(N^2 \log N)$ time!

Full Example: How “ring-like” are jets?

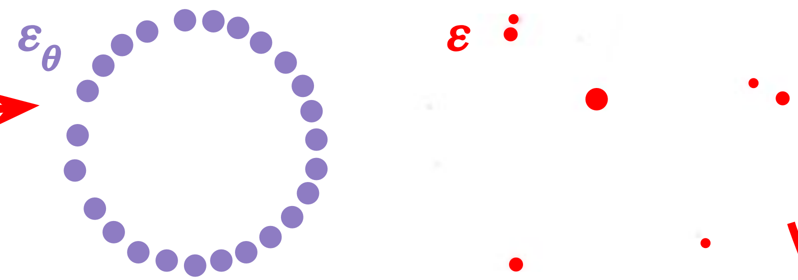
Step 1: Define the shape with parameters




```
def sample_circle(params, N, seed):

    thetas = jax.random.uniform(seed, shape
    x = params["Radius"] * jnp.cos(thetas)
    y = params["Radius"] * jnp.sin(thetas)
```

Unlike ordinary EMD, not necessary to specify center / orientation!

Step 2: Sample from Parameterized Shapes



Step 3: Calculate the spectral metric between events and shapes

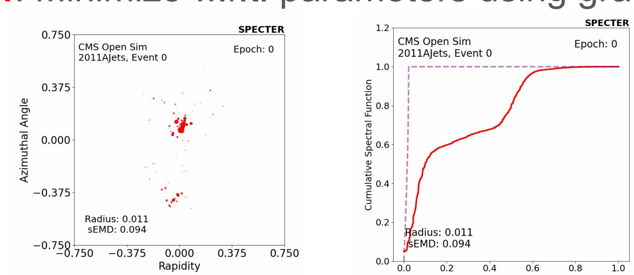
$$\mathrm{SEMD}_{\beta,p=2}(s_A, s_B) = \sum_{i<j\in\mathcal{E}_A} 2E_iE_j\omega_{ij}^2 + \sum_{i<j\in\mathcal{E}_B} 2E_iE_j\omega_{ij}^2$$
$$-\,2\sum_{n\in\mathcal{E}_A^2,\, l\in\mathcal{E}_B^2} \omega_n\omega_l \left(\min\left[S_A(\omega_n^+), S_B(\omega_l^+)\right] - \max\left[S_A(\omega_n^-), S_B(\omega_l^-)\right]\right)$$
$$\times\, \Theta\left(S_A(\omega_n^+) - S_B(\omega_l^-)\right)\Theta\left(S_B(\omega_l^+) - S_A(\omega_n^-)\right),$$

Key difference from previous work: We use the SEMD, *not* the EMD!

We have an explicit formula for the spectral EMD, so we can automatically differentiate through it

Standard ML procedure: Sample, calculate gradients, gradient descent, repeat! Analogous to WGANS.

Step 4: Minimize w.r.t. parameters using grads

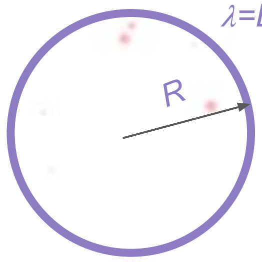


Pictured: Animation of optimizing for the radius R

Full Example: How “ring-like” are jets?

SPECTER Our code framework for these calculations

Step 1: Define the shape with parameters



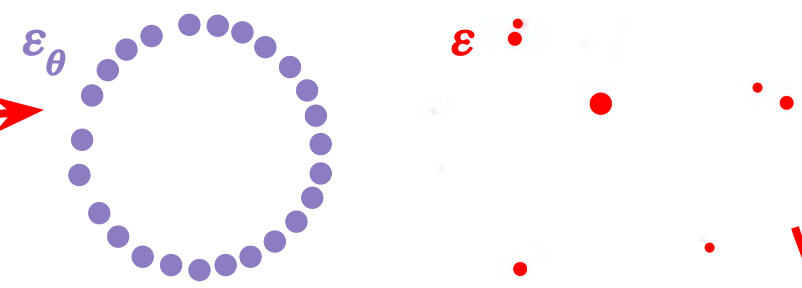

```python
def sample_circle(params, N, seed):

    thetas = jax.random.uniform(seed, shape
    x = params["Radius"] * jnp.cos(thetas)
    y = params["Radius"] * jnp.sin(thetas)
```

Unlike ordinary EMD, not necessary to specify center / orientation!

SPECTER is our code interface for performing these steps: sampling from user-defined shapes, calculating spectral functions and differentiable EMDS, and optimizing over parameters.

Built in highly parallelized and compiled JAX

Step 2: Sample from Parameterized Shapes

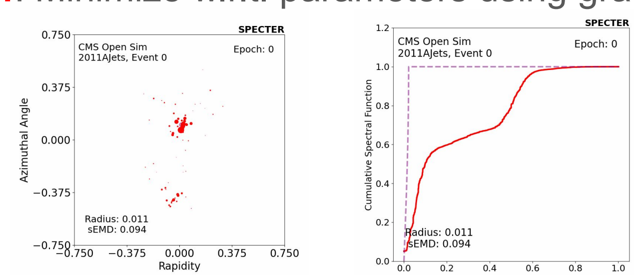


Step 3: Calculate the spectral metric between events and shapes

$$\mathrm{SEMD}_{\beta,p=2}(s_A,s_B) = \sum_{i<j\in\mathcal{E}_A} 2E_iE_j\omega_{ij}^2 + \sum_{i<j\in\mathcal{E}_B} 2E_iE_j\omega_{ij}^2$$
$$- 2\sum_{n\in\mathcal{E}_A^2,\, l\in\mathcal{E}_B^2} \omega_n\omega_l \left(\min\left[S_A(\omega_n^+), S_B(\omega_l^+)\right] - \max\left[S_A(\omega_n^-), S_B(\omega_l^-)\right]\right)$$
$$\times\, \Theta\left(S_A(\omega_n^+) - S_B(\omega_l^-)\right)\Theta\left(S_B(\omega_l^+) - S_A(\omega_n^-)\right),$$

Key difference from previous work: We use the SEMD, *not* the EMD!

Step 4: Minimize w.r.t. parameters using grads



Pictured: Animation of optimizing for the radius R

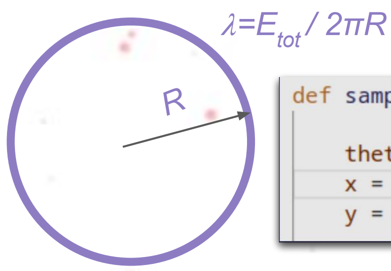
SPECTER is a “sequel” to SHAPER, introduced last ML4Jets. SPECTER is *not* an acronym, don’t ask me what it stands for.

Full Example: How "ring-like" are jets?

SPECTER

Our code framework for these calculations

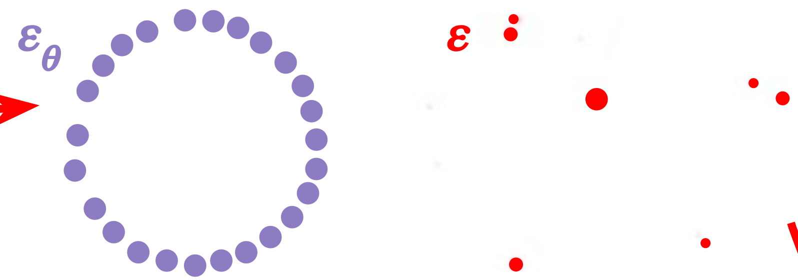
Step 1: Define the shape with parameters




```
def sample_circle(params, N, seed):

    thetas = jax.random.uniform(seed, shape
    x = params["Radius"] * jnp.cos(thetas)
    y = params["Radius"] * jnp.sin(thetas)
```

Unlike ordinary EMD, not necessary to specify center / orientation!

Step 2: Sample from Parameterized Shapes

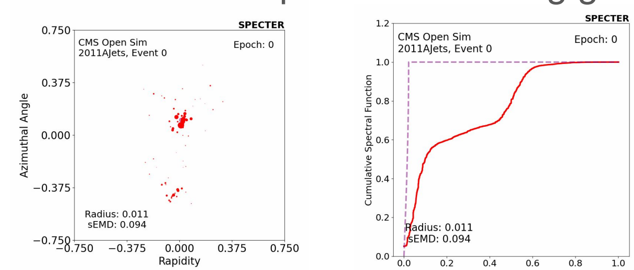


Step 3: Calculate the spectral metric between events and shapes

$$
\begin{aligned}
\mathrm{SEMD}_{\beta,p=2}(s_A, s_B) &= \sum_{i<j\in\mathcal{E}_A} 2E_iE_j\omega_{ij}^2 + \sum_{i<j\in\mathcal{E}_B} 2E_iE_j\omega_{ij}^2 \\
&- 2\sum_{n\in\mathcal{E}_A^2,\, l\in\mathcal{E}_B^2} \omega_n\omega_l \left(\min\left[S_A(\omega_n^+), S_B(\omega_l^+)\right] - \max\left[S_A(\omega_n^-), S_B(\omega_l^-)\right]\right) \\
&\times \Theta\left(S_A(\omega_n^+) - S_B(\omega_l^-)\right)\Theta\left(S_B(\omega_l^+) - S_A(\omega_n^-)\right),
\end{aligned}
$$

Key difference from previous work: We use the SEMD, *not* the EMD!

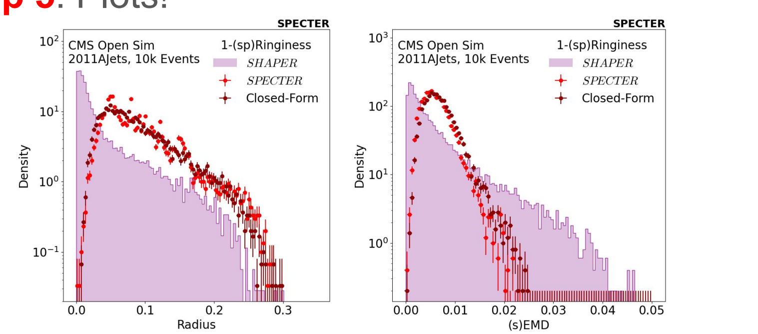
Step 4: Minimize w.r.t. parameters using grads



Pictured: Animation of optimizing for the radius R

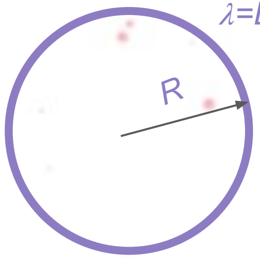
Pictured: 10k Jets, CMS 2011AJets Open Sim

Step 5: Plots!



Full Example: How "ring-like" are jets?

Step 1: Define the shape with parameters



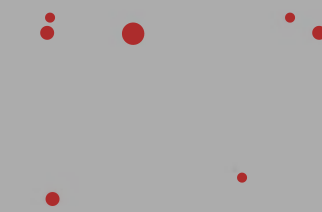

```
def sampl
    theta
    x = p
    y = p
```

Unlike ordinary EMD, not necessary to s

SPECTER

Our code framework for these calculations

Step 2: Sample from Parameterized Shapes



Alternatively…

The spectral EMD, *and its optimization*, are often partially or ***completely solvable*** in closed form!

$$R_{\text{opt}} = \frac{2}{\pi} \sum_{\substack{n\in\mathcal{E}^2 \\ \omega_n<\omega_{n+1}}} \omega_n \left[\sin\left(\frac{\pi}{2E_{\text{tot}}^2} \sum_{\substack{n\le m\in\mathcal{E}^2 \\ \omega_m<\omega_{m+1}}} (2EE)_m\right) - \sin\left(\frac{\pi}{2E_{\text{tot}}^2} \sum_{\substack{n<m\in\mathcal{E}^2 \\ \omega_m<\omega_{m+1}}} (2EE)_m\right)\right]$$

$$\text{SEMD}_{\beta,p=2}(s, s_{\text{jet ring}}) = \sum_{i<j\in\mathcal{E}} 2E_iE_j\omega_{ij}^2 - 2E_{\text{tot}}^2 R_{\text{opt}}^2$$

For many shapes, we can completely short circuit having to perform expensive optimization over an optimal transport problem entirely!

e spectral metric between

$$\sum_{\mathcal{E}_A} 2E_iE_j\omega_{ij}^2 + \sum_{i<j\in\mathcal{E}_B} 2E_iE_j\omega_{ij}^2$$

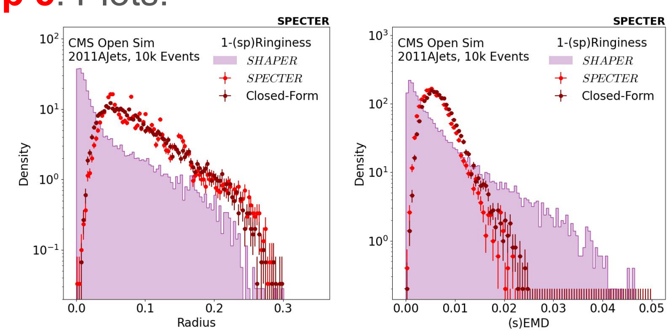
$$(\omega_n^+), S_B(\omega_l^+)] - \max\left[S_A(\omega_n^-), S_B(\omega_l^-)\right])$$

$$- S_B(\omega_l^-))\,\Theta\left(S_B(\omega_l^+) - S_A(\omega_n^-)\right),$$

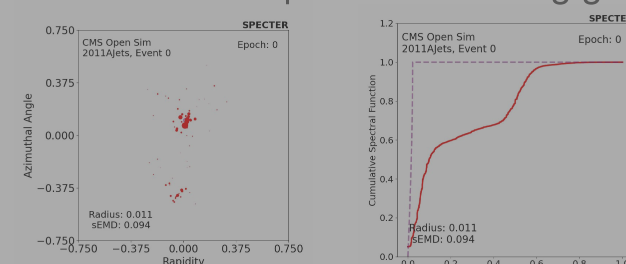
Key difference from previous work: We use the SEMD, *not* the EMD!

Pictured: 10k Jets, CMS 2011AJets Open Sim

Step 5: Plots!



Step 4: Minimize w.r.t. parameters using grads



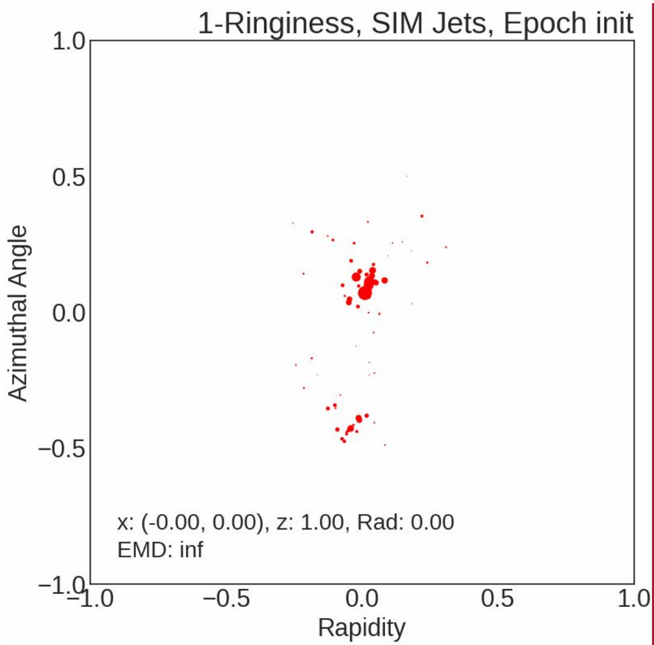
Pictured: Animation of optimizing for the radius R

[1][Ba, Dogra, **RG**, Tasissa, Thaler, 2302.12266]

To distinguish SEMD observables from EMD observables, I will add “s” or “sp”

Hearing Jets (sp)Ring

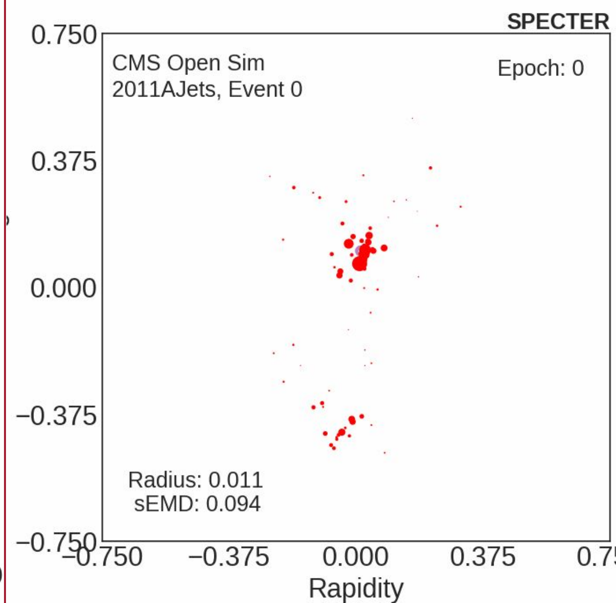
EMD

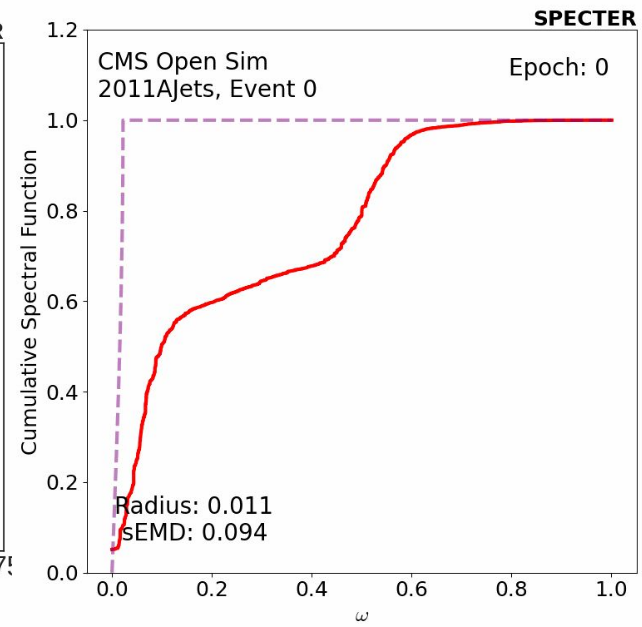


Calculated using SHAPER[1]

Position of ring must be optimized – can use as jet algorithm

Spectral EMD





Calculated using ***SPECTER***

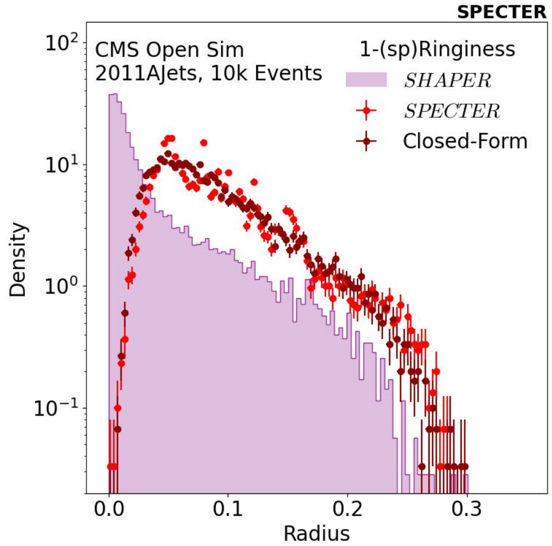
Translationally invariant – no need to optimize over position

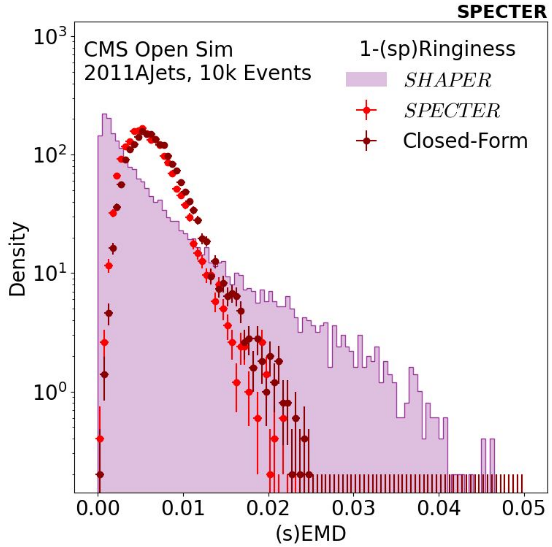
Secretly a 1D optimal transport problem over the spectral function

Hearing Jets (sp)Ring

Runtimes (Laptop CPU 12th Gen Intel(R) Core(TM) i7-1255U):
SHAPER (EMD): ~ 3 hours / 10k events
Generalized *SPECTER*: ~15 minutes / 10k events
Closed Form *SPECTER*: ~3 seconds / 10k events

The SEMD and EMD give qualitatively different radii! We can try to use our expression for R_{opt} to do fixed-order calculations to try to understand the the SEMD result:





Hearing Jets (sp)Ring

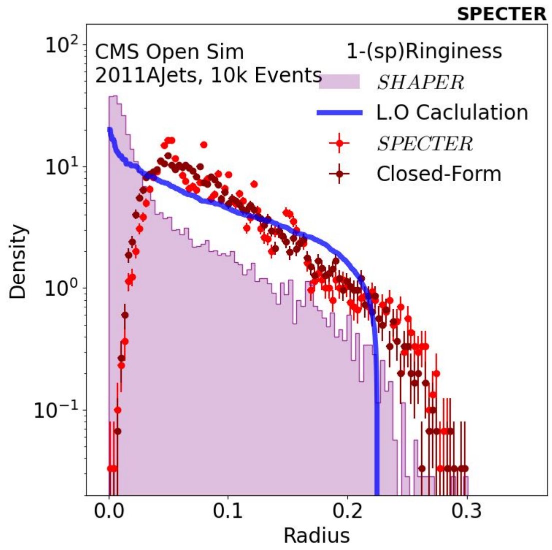
Runtimes (Laptop CPU 12th Gen Intel(R) Core(TM) i7-1255U):
SHAPER (EMD): ~ 3 hours / 10k events
Generalized *SPECTER*: ~15 minutes / 10k events
Closed Form *SPECTER*: ~3 seconds / 10k events

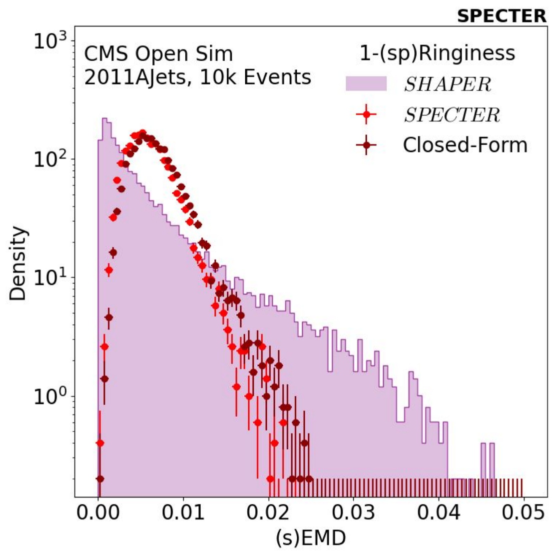
The SEMD and EMD give qualitatively different radii! We can try to use our expression for R_{opt} to do fixed-order calculations to try to understand the the SEMD result:

$$R_{\text{opt}} = \frac{2}{\pi}\omega \sin\left(z(1-z)\pi\right)$$

$$\frac{d\sigma^{\text{LO}}}{dR_{\text{opt}}} \sim \frac{\alpha_S}{\pi}\sum_{i=q,g} f_i \int_0^R \frac{d\theta}{\theta}\int_0^1 dz\, P_i(z)\delta(R_{\text{opt}} - R_{\text{opt}}(z,\theta))$$

quark/gluon fraction   quark/gluon splitting function





Hearing Jets (sp)Ring

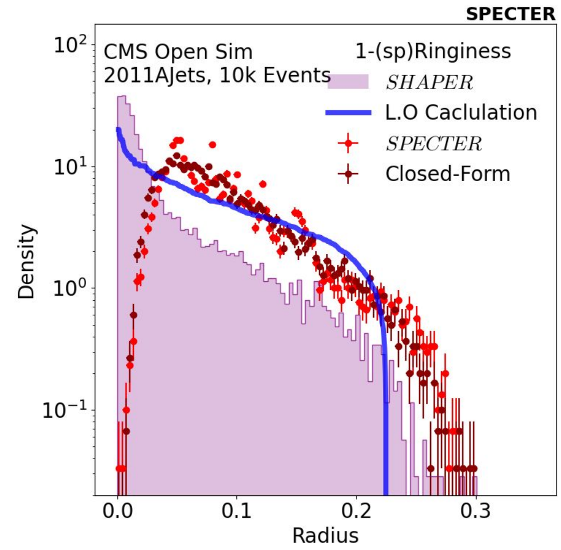
Runtimes (Laptop CPU 12th Gen Intel(R) Core(TM) i7-1255U):
SHAPER (EMD): ~ 3 hours / 10k events
Generalized *SPECTER*: ~15 minutes / 10k events
Closed Form *SPECTER*: ~3 seconds / 10k events

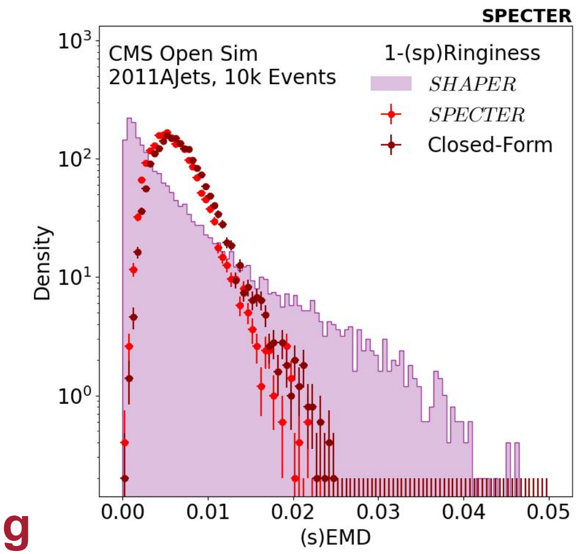
The SEMD and EMD give qualitatively different radii! We can try to use our expression for R_{opt} to do fixed-order calculations to try to understand the the SEMD result:

$$R_{\rm opt} = \frac{2}{\pi}\omega \sin\left(z(1-z)\pi\right)$$

$$\frac{d\sigma^{\rm LO}}{dR_{\rm opt}} \sim \frac{\alpha_S}{\pi}\sum_{i=q,g} f_i \int_0^R \frac{d\theta}{\theta}\int_0^1 dz\, P_i(z)\delta(R_{\rm opt} - R_{\rm opt}(z,\theta))$$

quark/gluon fraction   quark/gluon splitting function





It's possible to gain a first-principles understanding of these ML-inspired observables!

Things to think about:

- The current implementation of SPECTER, shown today takes $O(N^2)$ more memory than it ideally needs to, but this is not a fundamental issue and can be solved by staring at JAX documentation for even longer.
- For pairs of events with just a few particles, the SEMD and EMD^2 agree exactly before degenerate points in phase space – can we identify precisely when this happens?
- Not every shape has a completely closed-form solution, but it is usually possible to partially simplify and reduce the problem to 1D minimization, 1D root finding, or simple 1D numeric integrals.
- Closed-form and simple expressions means perturbative calculations may be possible – can we predict the radius of a jet?

Conclusion

The **spectral EMD** can be used as an alternative to, or an approximation of, the EMD. It is **fast** and **easy to minimize**.

SPECTER is a code package for efficiently evaluating the spectral EMD and calculating shape observables.

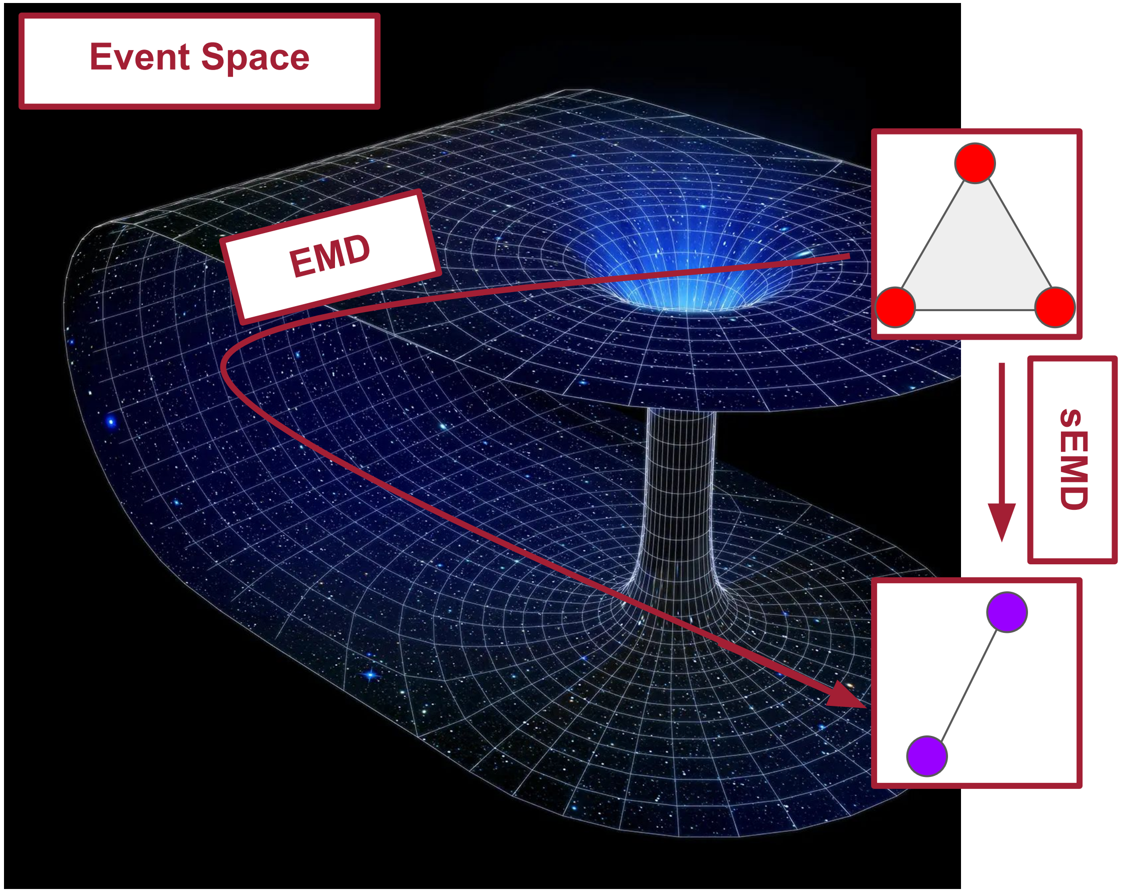
With the spectral EMD, many jet observables can be understood in **closed form**.

SPECTER

More questions? Email me at rikab@mit.edu

Appendices

Degeneracies

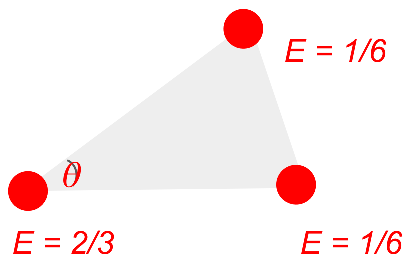


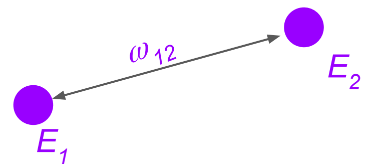
Highly symmetric configurations have degenerate spectral functions!

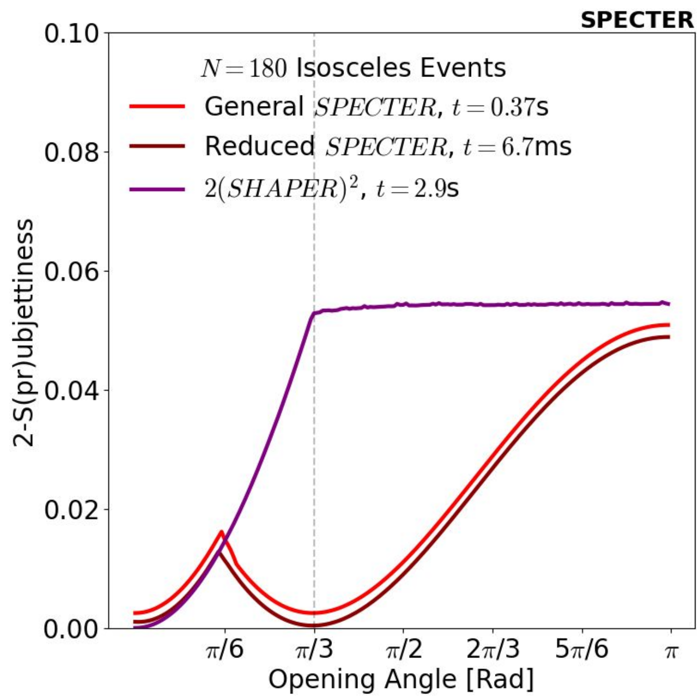
e.g. Equilateral Triangles* "look like" 2 particle events in their spectral functions!

*with the right energy weights.

Degeneracies (Continued)







For this precise energy configuration, equilateral triangles are *exactly* degenerate with 2 particle events – so the spectral EMD only sees 2 particles!

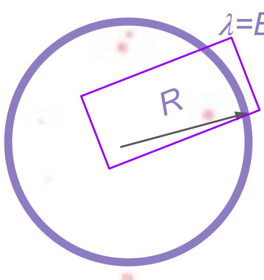
Only measure 0 configuration of events – but events *near* this give spectral EMDs *near* zero against 2 particle events.

*with the right energy weights.

Full Example: How "ring-like" are jets?

SPECTER Our code framework for these calculations

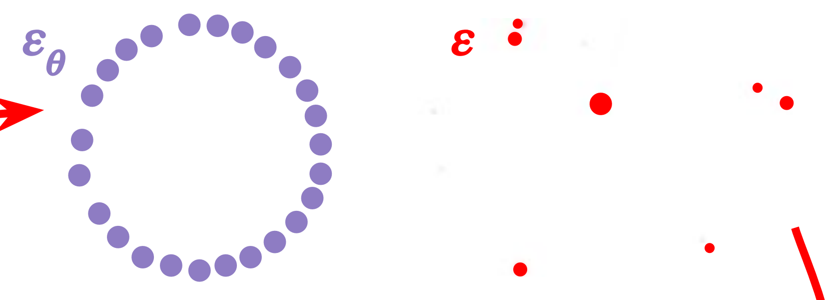
Step 1: Define the shape with parameters




```python
def sample_circle(params, N, seed):

    thetas = jax.random.uniform(seed, shape
    x = params["Radius"] * jnp.cos(thetas)
    y = params["Radius"] * jnp.sin(thetas)
```

Unlike ordinary EMD, not necessary to specify center / orientation!

Step 2: Sample from Parameterized Shapes

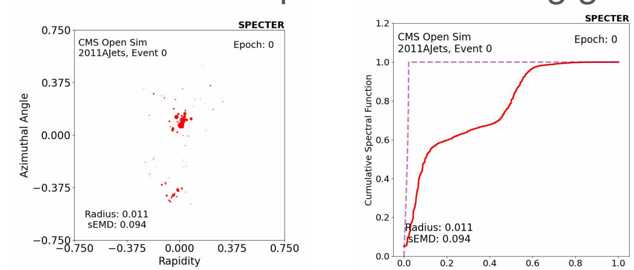


Step 3: Calculate the spectral metric between events and shapes

$$\mathrm{SEMD}_{\beta,p=2}(s_A, s_B) = \sum_{i<j\in\mathcal{E}_A} 2E_iE_j\omega_{ij}^2 + \sum_{i<j\in\mathcal{E}_B} 2E_iE_j\omega_{ij}^2$$
$$-2\sum_{n\in\mathcal{E}_A^2,\, l\in\mathcal{E}_B^2} \omega_n\omega_l \left(\min\left[S_A(\omega_n^+), S_B(\omega_l^+)\right] - \max\left[S_A(\omega_n^-), S_B(\omega_l^-)\right]\right)$$
$$\times\, \Theta\left(S_A(\omega_n^+) - S_B(\omega_l^-)\right)\Theta\left(S_B(\omega_l^+) - S_A(\omega_n^-)\right),$$

Key difference from previous work: We use the SEMD, *not* the EMD!

Step 4: Minimize w.r.t. parameters using grads



Pictured: Animation of optimizing for the radius R

Pictured: 10k Jets, CMS 2011AJets Open Sim

Step 5: Plots!