Bootstrapping String-Like Models using Entanglement Minimization and Machine Learning

Work with: Debapriyo Chowdhury, Arnab Priya Saha, Aninda Sinha

Based on arXiv:2409.18259 [hep-th]

Cosmological Correlators Worshop, LeCosPA, NTU, Taiwan, 2024

Faizan Bhat, Centre for High-Energy Physics,
Indian Institute of Science, Bangalore, India

Open String Amplitude

- **Open String Amplitude**: Describes the tree-level, two-two scattering of open superstrings ($\alpha' = \ell_s^2 = 1$)

$$\boxed{\mathcal{M}^{(OS)}(s,t) = \frac{\Gamma(-\alpha' s)\Gamma(-\alpha' t)}{\Gamma(1-\alpha' s-\alpha' t)}}$$

- **Duality**: For $Re(s) < 1$,

$$\boxed{\mathcal{M}^{(OS)}(s,t) = \frac{1}{st} + \sum_{n=1}^{\infty} \frac{(s+1)(s+2)...(s+n-1)}{n!} \frac{1}{t-n}}$$

- In a (now-very-famous) paper, A.Sinha, A.P.Saha presented

$$\boxed{\begin{aligned} \frac{\Gamma(-s)\,\Gamma(-t)}{\Gamma(1-s-t)} = \frac{1}{st} + \sum_{n=1}^{\infty} \frac{1}{n!}\left(\frac{1}{s-n} + \frac{1}{t-n} + \frac{1}{\lambda+n}\right) \\ \times \left(1-\lambda+\frac{(s+\lambda)(t+\lambda)}{\lambda+n}\right)_{n-1} \end{aligned}}$$

Field Theory Representations

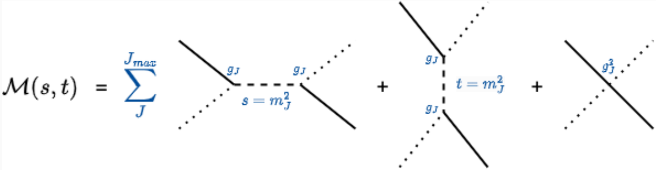

Figure 1: Field Theory Representation: Poles in all channels + contact terms

- Lorentz invariance implies a partial wave expansion of the residues:

$$\mathrm{Res}_{s=n}\mathcal{M}(s,t) = -\pi \sum_{\ell} c_\ell^{(n)} \mathcal{C}_\ell^{\frac{D-3}{2}}\left(z = 1 + \frac{2t}{s}\right)$$

where $\mathcal{C}_\ell^{\frac{D-3}{2}}(z)$ are the Gegenbauer polynomials in D dimensions.

- The general formula looks like

$$\boxed{\mathcal{M}(s,t) = \sum_{n=1}^{\infty}\sum_{\ell=0}^{\ell_{max}}\left[\frac{1}{s-n} + \frac{1}{t-n} + \frac{1}{\lambda+n}\right] c_\ell^{(n)}\, \mathcal{C}_\ell^{\left(\frac{D-3}{2}\right)}\left[1 + \frac{2}{n}\left(\frac{(s+\lambda)(t+\lambda)}{\lambda+n} - \lambda\right)\right]}$$

S-Matrix Bootstrap

- Bootstrap Approach: Find the allowed space of scattering amplitudes by imposing physical constraints.
- Wilson Coefficients: Expansion around $s+t=0$ and $s\ t=0$

$$\mathcal{M}_{low}(s,t) = W_{00} + W_{10}(s+t) + W_{01}st + \cdots$$

- **Q: In the space of consistent scattering amplitudes that satisfy duality, is the open superstring amplitude special?**
- **A: Yes, it minimizes the total entanglement generated in the scattering process.**
- **Measured by** $\sim -\mathbf{W_{0,0}}$. [Aoude, Elor, Remmen, Sumensari]
- Bootstrap Constraints: In $D=10$,
  - Crossing Symmetry: $\mathcal{M}(s,t) = \mathcal{M}(t,s)$
  - Analyticity: Only simple poles at $s=n, \forall n \in \mathbb{Z}_{\geq 0}$.
  - Residues at Poles: Polynomials of order $\ell_{max} = n-1$.
  - Fix $W_{1,0} = \zeta(3)$ and $W_{0,1} = \frac{7}{4}\zeta(4)$, i.e. open string values.
  - Unitarity: At tree-level, $c_\ell^{(n)} \geq 0$
  - λ-Independence: $\partial_\lambda^k \mathcal{M}_\lambda(s,t) = 0$, for $k \in \mathbb{Z}_{\geq 1}, \lambda \geq -1, (s,t) \in \mathcal{D}_\lambda$

S-Matrix Bootstrap

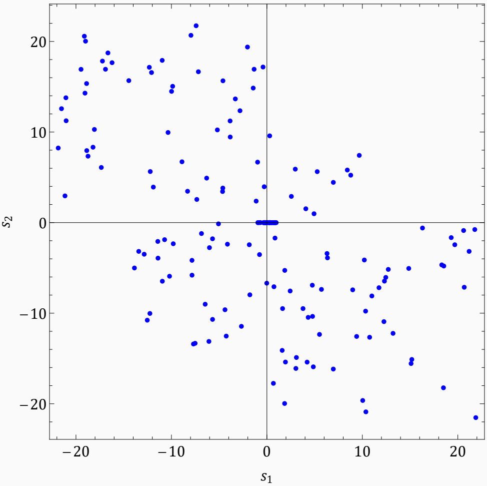

Figure 2: Example of $\mathcal{D}_\lambda$ we use for bootstrap

S-Matrix Bootstrap

- **Maximize $\mathbf{W_{0,0}}$ / Minimize Entanglement** (via SDPB)
  $(N_{max} = 30, \epsilon = 10^{-9}, k_{max} = 6, \lambda = 14.6)$

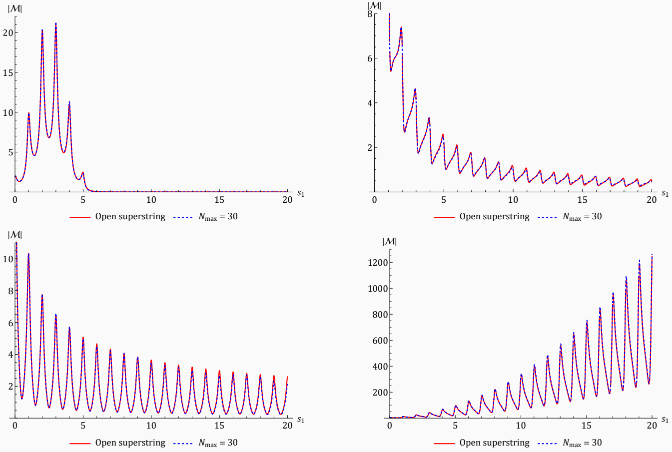

Figure 3: We plot $\left|\mathcal{M}\left(s + \frac{i}{10}, t\right)\right|$ versus s for $t = \{-5.1, -0.1, 0.5, 3.2\}$.

S-Matrix Bootstrap

- **Maximize $\mathbf{W_{0,0}}$ / Minimize Entanglement** (via SDPB)
  $(N_{max} = 30, \epsilon = 10^{-9}, k_{max} = 6, \lambda = 14.6)$

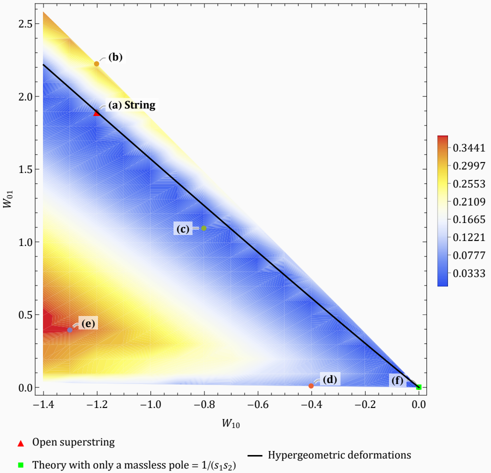

Figure 4: Space of Open-String like theories

Bootstrap using PINNs

- **Why do we use PINNs for bootstrap?**
- The bootstrap problems we discussed till now and in fact, all bootstrap problems are either Linear optimization problems, or Semi-definite optimization problems.
- These can be handled extremely well via traditional methods.
- Caveat: We used $\epsilon \sim 10^{-9}$ while imposing the constraint

$$-\epsilon \leq \partial_\lambda^k \mathcal{M}_\lambda \left(s, t, c_\ell^{(n)}\right) \leq \epsilon, \text{ for } (s,t) \in \mathcal{D}_\lambda \text{ and } 1 \leq k \leq k_{max}$$

  However, truncated to some N_{max}, it is not guaranteed that these constraints will be satisfied to some $\epsilon << 1$.
- It is more reasonable to impose ratio constraints

$$\left|1 - \frac{\mathcal{M}_{\lambda_1}(s,t)}{\mathcal{M}_{\lambda_2}(s,t)}\right| \leq \epsilon, \qquad \left|\frac{\partial_\lambda^k \mathcal{M}_\lambda(s,t)}{\mathcal{M}_\lambda(s,t)}\right| \leq \epsilon$$

- These are non-linear in the parameters $c_\ell^{(n)}$. Traditional methods like SDPB are not useful. This is why we use PINNs.

Bootstrap using PINNs

- Neural networks: Maps with several tunable parameters. In our case,

$$\mathrm{NN}\left(\ell, n, \theta_j\right) \equiv c_\ell^{(n)}$$

- Neuron Input-Output: $y_j^M = \sigma\left(\sum_{k=1}^{k_{max}} w_{j,k}^M y_k^{M-1} + b_j^M\right)$

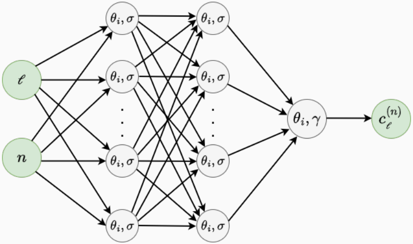

Figure 5: Architecture: Input layer with 2 neurons for $(\ell,\ n)$, 2 hidden layers with 64 neurons, output layer wih 1 neuron for $c_\ell^{(n)}$. Every neuron has the ReLU activation function $\sigma(x) = \max(0, x)$. Final layer has the SoftPlus activation function $\gamma(x) = \log(1 + e^x)$ for positive $c_\ell^{(n)}s$.
Total Parameters $= [2(64) + 64] + [64(64) + 64] + [64(1) + 1] = 4417$.

Bootstrap using PINNs

- Neural networks by minimizing a loss function that measure the violation of constraints.
- We define the following loss function

$$\boxed{\begin{aligned}\mathcal{L}\left(\theta_j^M\right) &= -W_{0,0} + \beta_1 \Big(W_{10} - (-\zeta(3))\Big)^2 + \beta_2 \left(W_{01} - \frac{7\zeta(4)}{4}\right)^2 \\ &\quad + \beta_3 \frac{1}{N} \sum_{s,t \in \mathcal{D}_\lambda} \tilde{\mathcal{L}}(s,t).\end{aligned}}$$

- Hyperparameters β_i set the tolerance for constraint violation. Bigger β_i means smaller tolerance $\implies \min(EPM) = \min\left(\mathcal{L}\left(\theta_j^M\right)\right)$

Bootstrap using PINNs

- We implement PINN using the Python library PyTorch.
- **Case 1:** When $\tilde{\mathcal{L}}(s,t) = \left(1 - \frac{\mathcal{M}_{\lambda_1}(s,t)}{\mathcal{M}_{\lambda_2}(s,t)}\right)^2$,
  - $\mathcal{D}_\lambda = \{(s,t)\,|\, -5.5 \le s \le 5.5\,, -0.2 \le t \le 0.2\,, \Delta_s = 1, \Delta_t = 0.4\}$
  - **Leading Regge Trajectory**:

	$c_0^{(1)}$	$c_1^{(2)}$	$c_2^{(3)}$	$c_3^{(4)}$	$c_4^{(5)}$	$c_5^{(6)}$
Open String	1	0.0714	0.0119	0.00289	0.000867	0.000300
PINN	0.999	0.0715	0.0121	0.00300	0.000922	0.000332

- **Case 2:** When $\tilde{\mathcal{L}}(s,t) = \sum_{k=1}^{k_{max}} \left(\frac{1}{\mathcal{M}_\lambda(s,t)} \frac{\partial^k \mathcal{M}_\lambda(s,t)}{\partial \lambda^k}\right)^2$,
  - $\mathcal{D}_\lambda = \{(s,t)\,|\, 0.4 \le s \le 10.4\,, t = 10.1\,, \Delta_s = 1\}$
  - **Leading Regge Trajectory**:

	$c_0^{(1)}$	$c_1^{(2)}$	$c_2^{(3)}$	$c_3^{(4)}$	$c_4^{(5)}$	$c_5^{(6)}$
Open String	1	0.0714	0.0119	0.00289	0.000867	0.000300
PINN	0.998	0.0702	0.0124	0.00341	0.000990	0.000339

  - For open string, at $(s,t) = (10.4, 10.1)$, $\mathcal{M}(s,t) \approx 1.34 \times 10^5$ and $\partial_\lambda \mathcal{M}_\lambda(s,t) \approx -2.51$. $\implies$ Only PINN method can work!.

Summary

- We present a new way to set up the numerical S-Matrix bootstrap using a parametric crossing symmetric dispersion relation $\lambda - CSDR$.
- We maximize $W_{0,0}$ / minimize the first finite moment of the entangling power (EPM) and find that the optimal solution is an excellent approximation to the open superstring amplitude.
- We initiate the use of Physics-Informed Neural Networks for the bootstrap to perform non-linear, constrained optimization.
- We also study closed string-like amplitudes and find Dual resonance models there also minimize EPM.

THANK YOU