Biasing optimization to find more informative model metamers

William F. Broderick[1], Erica Shook[2], Daniel Herrera-Esposito[3], and Eero P. Simoncelli[1,4]

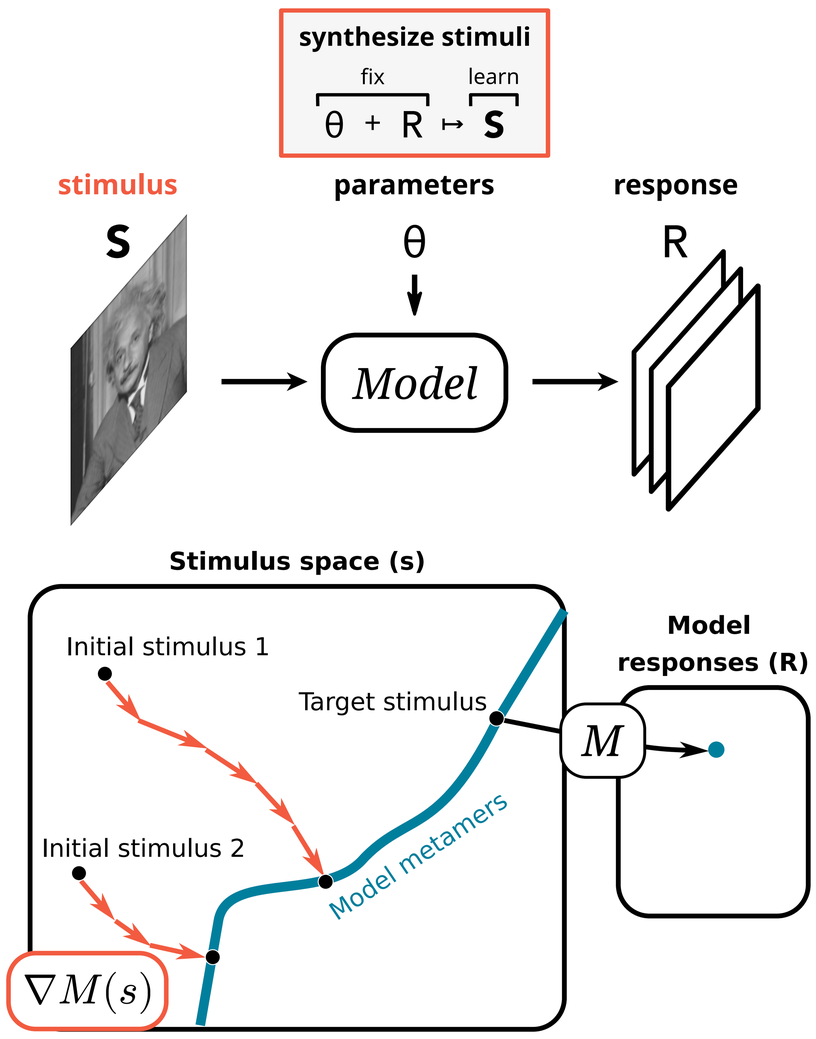
1. Flatiron Institute, Simons Foundation 2. Columbia University 3. Princeton University 4. New York University

Contact: wbroderick@flatironinstitute.org

CCN 2026

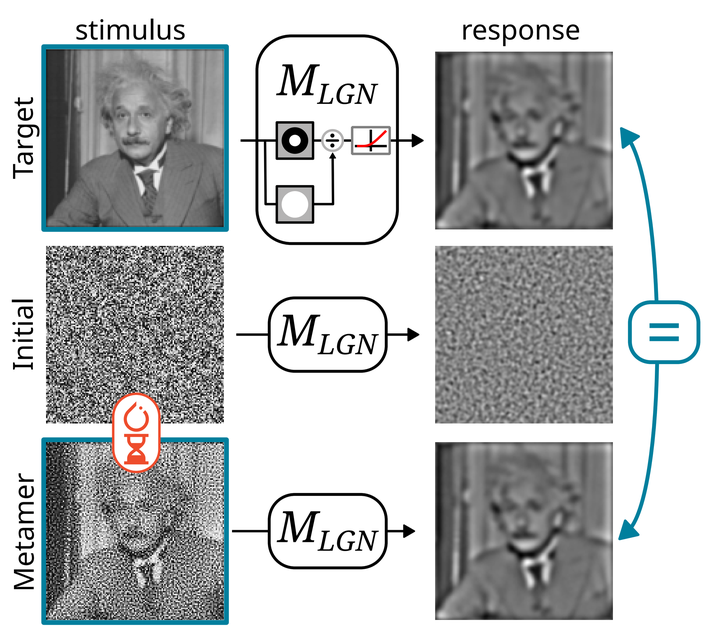
Model metamers

Images that are physically different but perceptually identical (according to a model)



Example model metamers

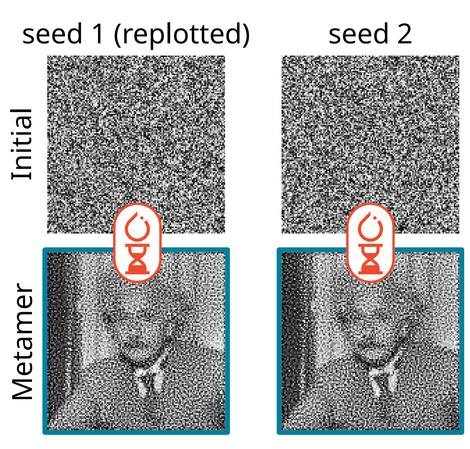
LGN-inspired model with gain control[1]



1. Berardino et al., 2017

How to explore metamer space?

LGN-inspired model metamers initialized with different patches of white noise look the same



Bias optimization with penalty term

Standard metamer synthesis:

$$\arg\min_{\hat{s}} L(M(s), M(\hat{s}))$$

L : loss function, e.g., mean-squared error

s : target stimulus tensor

$\hat{s}$: model metamer tensor

Add a penalty term*:

$$\arg\min_{\hat{s}} L(M(s), M(\hat{s})) + \lambda P(\hat{s})$$

λ : tradeoff

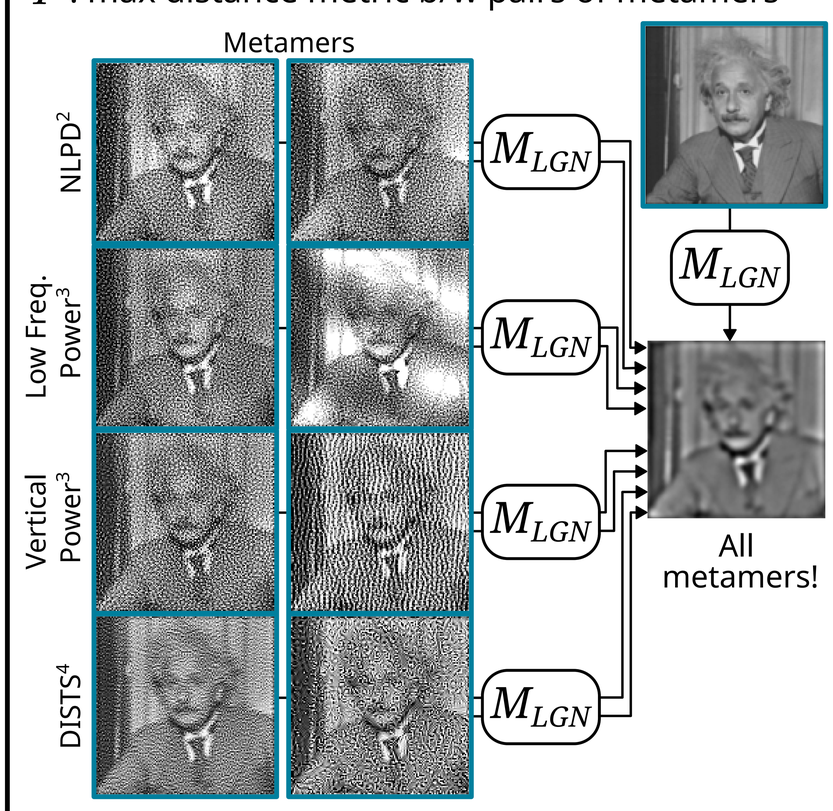
P: penalty function, e.g., range penalty

* Often works better in multiple stages

Example 1: increase visual diversity

LGN-inspired model with gain control

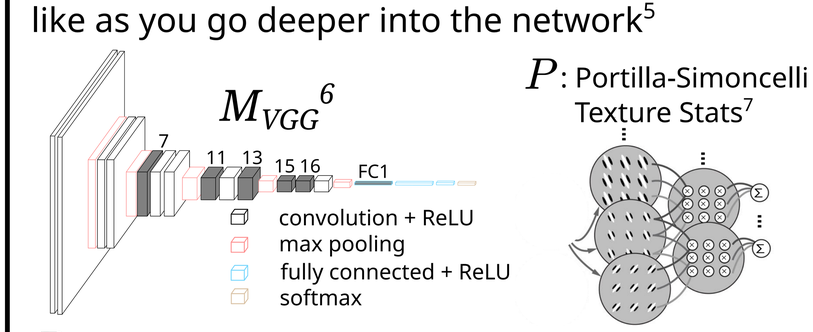
P: max distance metric b/w pairs of metamers



2. Normalized Laplacian Pyramid Distance, Laparra et al., 2016
3. Steerable Pyramid, Simoncelli and Freeman, 1995
4. Deep Image Structure and Texture Similarity, Ding et al., 2020

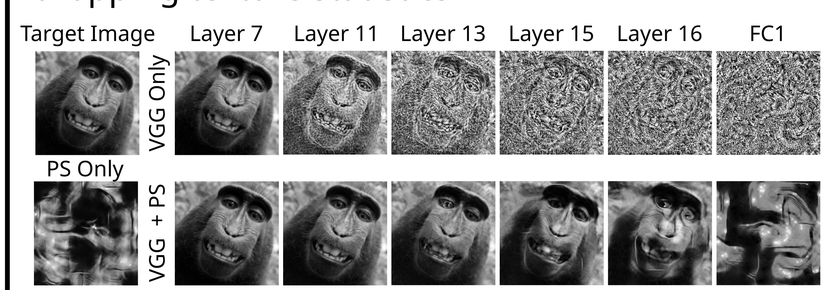
Example 2: joint model metamers

Standard deep net metamers become noise-like as you go deeper into the network[5]



P: minimize difference in PS stats b/w target image and metamer

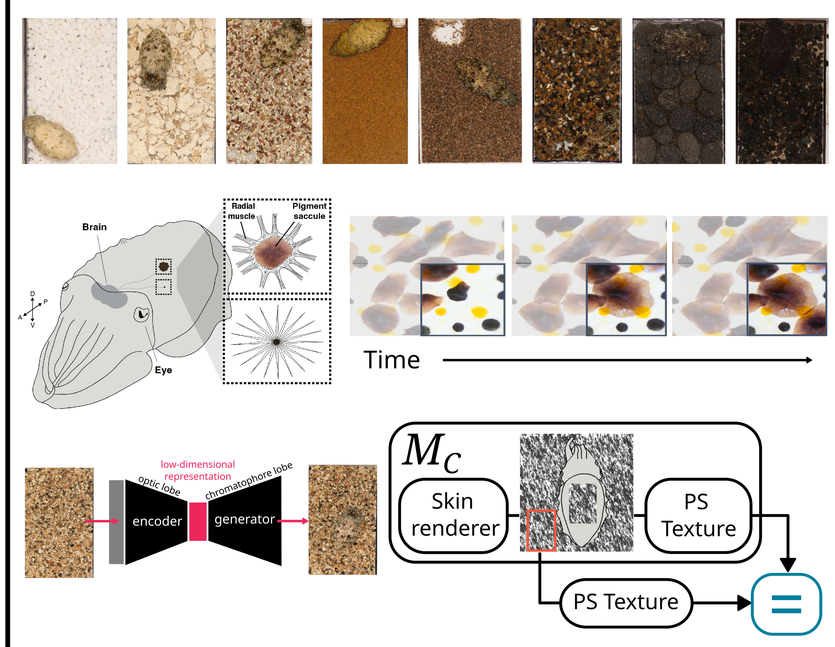
Adding PS constraint reveals deep nets are dropping texture statistics

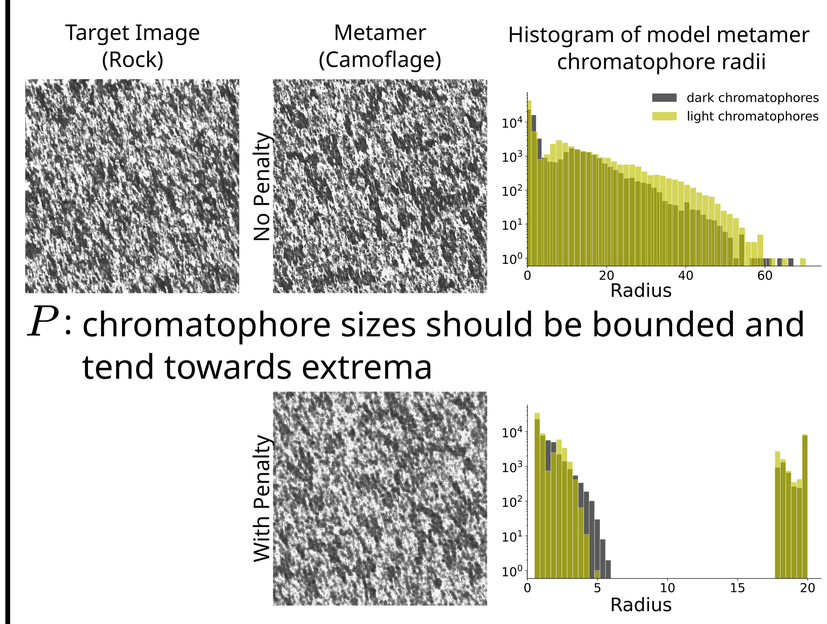


5. Feather et al., 2023 6. Simonyan and Zisserman, 2014 7. Portilla and Simoncelli, 1999

Example 3: impose biological constraints

Cuttlefish change color and skin pattern to avoid detection[8,9]





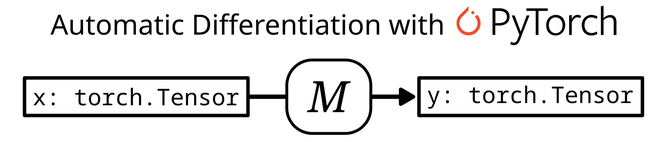
P: chromatophore sizes should be bounded and tend towards extrema

8. Shook et al., 2024
9. Shook et al., 2026

plenoptic

docs.plenoptic.org github.com/plenoptic-org/plenoptic

Automatic Differentiation with PyTorch



$\frac{\partial M}{\partial s}$ plenoptic can compute the gradients for **any model*** written in pytorch

* Using only operations PyTorch knows how to differentiate, i.e., using functions from the PyTorch library.

```
import plenoptic as po
img = po.data.einstein()
model = po.models.LuminanceGainControl(
    kernel_size=(31,31), pad_mode="circular",
    pretrained=True, cache_filt=True
).eval()
po.remove_grad(model)

def penalty(synthesized_image):
    # Compute something here
    ...
    return scalar

met = po.Metamer(
    img, model, penalty_function=penalty,
    penalty_lambda=0.02
)
met.synthesize(max_iter=1300, stop_criterion=1e-11)
```

load image
initialize and prepare model
define penalty (*optional*)
initialize metamer object
run optimization

More than images: video, audio
Compatible with any pytorch model: custom, torchvision, etc.
Synthesis methods: Metamers, Eigendistortions, MAD Competition
Visual Models: LGN-inspired, Texture, DeepNetFeatures
Image Quality Metrics: SSIM, MS-SSIM, NLPD
Canonical Computations: Laplacian Pyramid, Steerable Pyramid, Local gain control
Image processing tools: autocorrelations, up- and down-sampling, etc.