Emulation for Approximating Radiative Transfer Modeling: Computational Efficiency and Sensitivity Analysis

Jochem Verrelst, Emmanuel Johnson, Pablo Morcillo, Jorge Vicent, Juan Pablo Rivera Caicedo, and Gustau Camps-Valls

Any difference?   Which model would you choose?

37 min

0.2 s

500#

SCOPE (RTM)

Emulator (emulated SCOPE)

Advanced RTMs: generation of a large LUT (>1000#)

SCOPE   DART   MODTRAN

Hours   days   >days

Advanced RTMs: *more realistic but slow*

Emulation of RTMs

Emulators are statistical models that approximate the processing (input-output) of a physical model (e.g. RTM) - at a fraction of the computational cost:

making a statistical model from a physical model

RTM   Machine learning   Emulator

Regression vs. Emulation:

Spectra → regression / Emulation → Variables (e.g. LAI, chlorophyll...)

Common use of machine learning in optical RS:

Variable (e.g. LAI)

Statistical regression method:
- Variable/data-driven, black box, 1 output, portability is questionable

Emulation in optical RS:

Replace RTM:
- Multiple applications, e.g. inversion
  - ✓ Radiometric method: Spectral fitting
  - ✓ Portable: generally applicable

"spectral redundancy" is a blessing

Processing steps emulation

Input (variables + spectra) → Splitting into training/validation → PCA on spectra → MLRA training looping over components → Prediction of components → Reconstruction of spectra → Validation → Emulator

PCA on spectra

$$Sc = U \cdot X$$

MLRA training looping over components

$$W = (Y + \lambda I)^{-1} \cdot Sc$$

Prediction of components

$$Sp = Sc \cdot W$$

Reconstruction of spectra

$$Xr = U^T \cdot Sp$$

PCA components

Predicted spectra

Validation → Emulator

Emulator toolbox

ARTMO
http://ipl.uv.es/artmo/

With ARTMO's emulation processing chain any RTM can be converted into an emulator.

Input (variables + spectra) → Splitting into training/validation → PCA on spectra → MLRA training looping over components → Prediction of components → Reconstruction of spectra → Validation → Emulator

PCA

PLS-DA

$$Xr = U^T \cdot Sp$$

$$NRMSE = \frac{1}{p}\sum_{j=1}^{p}\sqrt{\frac{\sum_{i=1}^{n}(\hat{y}_i-y_i)^2}{\sum_{i=1}^{n}y_i}}$$

Latin Hypercube Sampling (LHS)
#1000

EMULATOR Toolbox [v. 1.08]
Input | Settings | Validation | Tools | Help | ^

Input | Settings | Validation | Emulator

Rivera, J.P.; Verrelst, J.; Gómez-Dans, J.; Muñoz-Marí, J.; Moreno, J.; Camps-Valls, G. (2015) *An Emulator Toolbox to Approximate Radiative Transfer Models with Statistical Learning*. Remote Sensing. p. 7, 9347-9370.

Emulators applied into GSA: PROSPECT-4

ARTMO's GSA toolbox

PROSPECT-4 (122 s)

X 2

KRR (55 s)   NN (55 s)   GPR (58 s)

1000#/variable

N   Cw   Cab   Cm

PROSAIL

4SAIL (103 s)   Emulator GPR (32s)

X 3

Emulator KRR (28s)   Emulator VHGPR (37s)

1000#/variable

N Cw Cab Cm LAI LAD soil coeff SZA

Coupling PROSAIL + MODTRAN: GPR emulator

$$L = L_0 + \frac{E_{tot}T_{tot}\rho}{\pi(1 - S\rho)}$$

PROSAIL + MODTRAN   GSA type:Saltelli / Subsamples:1000   RT model: Emulator / Sensor:NO SENSOR

N
Cab
Cw
Cm
LAI
LAD
O3STR
H2OSTR
AOT550
G
ASTMX
SSA

Sensitivity Analysis for GP Gradient-Based Methods for Emulation

Model: Gaussian Processes

Parameters: θ

Noise: σ_y^2

$\mathbf{x}$ Data → $\mathbf{f}$ Model → $\mathbf{y}$ Outputs

jemanjohnson34@gmail.com
jejjohnson.netlify.com
https://github.com/jejjohnson
https://github.com/IPL-UV/gp_sens
https://isp.uv.es

Gaussian Processes

GP Prior
$$f \sim \mathcal{GP}$$
$$f \sim \mathcal{N}(0, K)$$

Noisy Observations
$$y_i | f_i \sim \mathcal{N}(f_i, \sigma_i^2)$$

GP Posterior
$$Draws \sim P(f|Data)$$

Mean and Error Bars

The Derivative of Kernel Functions

Most Kernel methods can be written as: $\mathbf{k}\alpha$

$$\frac{\partial f(\mathbf{x}_*)}{\partial x^j} = \frac{\partial \mathbf{k}_* \alpha}{\partial x^j}$$

Example kernel function:
$$K(\mathbf{x}, \mathbf{y}) = \exp(-\gamma \|\mathbf{x} - \mathbf{y}\|)$$

Note: the only term that depends on x is k.

$$\frac{\partial f(\mathbf{x}_*)}{\partial x^j} = (\partial_j \mathbf{k}_*)^T \alpha$$

Final kernel method derivative:
$$\frac{\partial f(\mathbf{x})}{\partial x^j} = -2\gamma(x^j - y^j)\exp(-\gamma\|\mathbf{x} - \mathbf{y}\|) \cdot \alpha$$

GP methods

Training time: 160 sec

2.961 s   Gradient Sensitivity

12.956 s   Sobel Indices (Main Effects)

N
Cab
Cw
Cm
lai
angle
psoil
tts

Resources

Papers:
- Gaussian Processes
  - http://www.gaussianprocess.org/
  - Camps-Valls et al., *A Survey on Gaussian Processes for Earth-Observation Data Analysis: A Comprehensive Investigation*, IEEE GRSM, Volume 4, Issue 2, 2016
- Emulation with Gaussian Processes
  - Verrelst et al, *Global Sensitivity Analysis of Leaf-Canopy-Atmospheric RTMs: Implications for Biophysical Variables Retrieval from Top-of-Atmosphere Radiance Data*, MDPI, 11 (16), 2019

Software
- GPy - Academic GPs
  - https://github.com/SheffieldML/GPy
- Scikit-Learn - Standard GPs
  - https://scikit-learn.org/
- EmuKit - Sensitivity Analysis
  - https://github.com/emukit/
- Demonstration with Prosail
  - https://github.com/IPL-UV/gp_sens

Emulators into numerical inversion

Image

Spectral fitting:
Minimization algorithm: lsqnonlin

Output maps of RTM variables

RTM

Per-pixel RTM iterations: very slow method, inapplicable to advanced RTMs.

Emulation of an RTM

Forest
DART KRR emulator applied to HyPlant DUAL (450-2500 nm)

Agriculture
SCOPE KRR emulator applied HyPlant DUAL (sub-surface added)

Emulation for synthetic scene generation.

Emulation of synthetic hyperspectral image

S2   Hyperspectral S2 (125 bands)
KRR HyMap Emulation
< 1 min

A S2 image is used as input to emulate an hyperspectral image with a S2 texture.

Conclusions

- Emulation of RTMs: physically-aware usage of machine learning, e.g. Gaussian processes (GP)

- Emulators approximate RTM outputs at a fraction of the computational costs

- Emulators suitable for global sensitivity analysis of advanced or coupled RTMs

- With GP sensitivity indices also achieved through derivatives

- Emulation attractive for multiple EO applications: e.g., inversion, scene generation