L2 Retrieval Efforts within the U.S. Greenhouse Gas Center

Peter Somkuti – peter.somkuti@nasa.gov
ESSIC / NASA GMAO
Lesley Ott – lesley.ott@nasa.gov
NASA GMAO

GODDARD EARTH SCIENCES

NASA

Contents

- U.S. Greenhouse Gas Center
- Identifying Needs in Retrieval Algorithm Development
  - How do we improve on the state-of-the-art?
- Why are GHG Retrieval Algorithms so Complicated?
  - Answer: because we want choices
- Providing new tools to improve accessibility
  - Focus: documentation
- Summary and Outlook

GODDARD EARTH SCIENCES

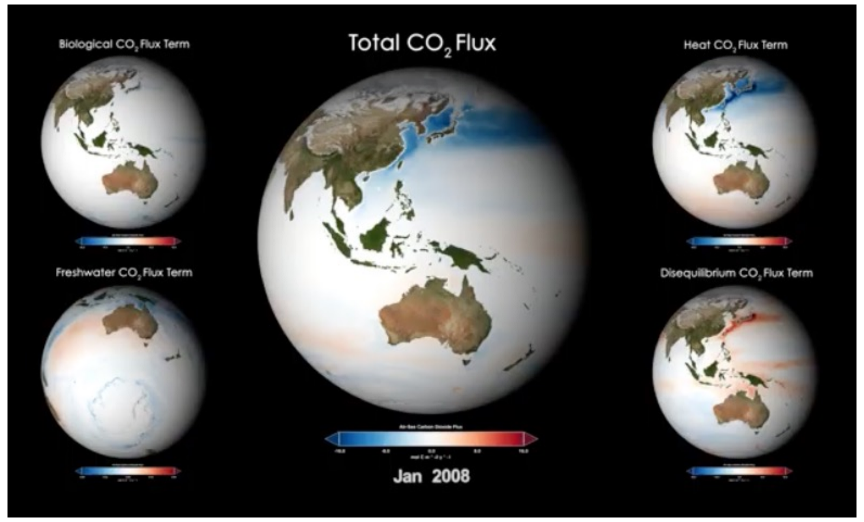
Supported by U.S. Greenhouse Gas Center

- https://earth.gov/ghgcenter

- Multi-agency collaboration to consolidate GHG information from models and observations

- Website features a data hub, exploration and analysis platform

NASA · United States Environmental Protection Agency · NIST National Institute of Standards and Technology U.S. Department of Commerce · NOAA



As part of accompanying efforts

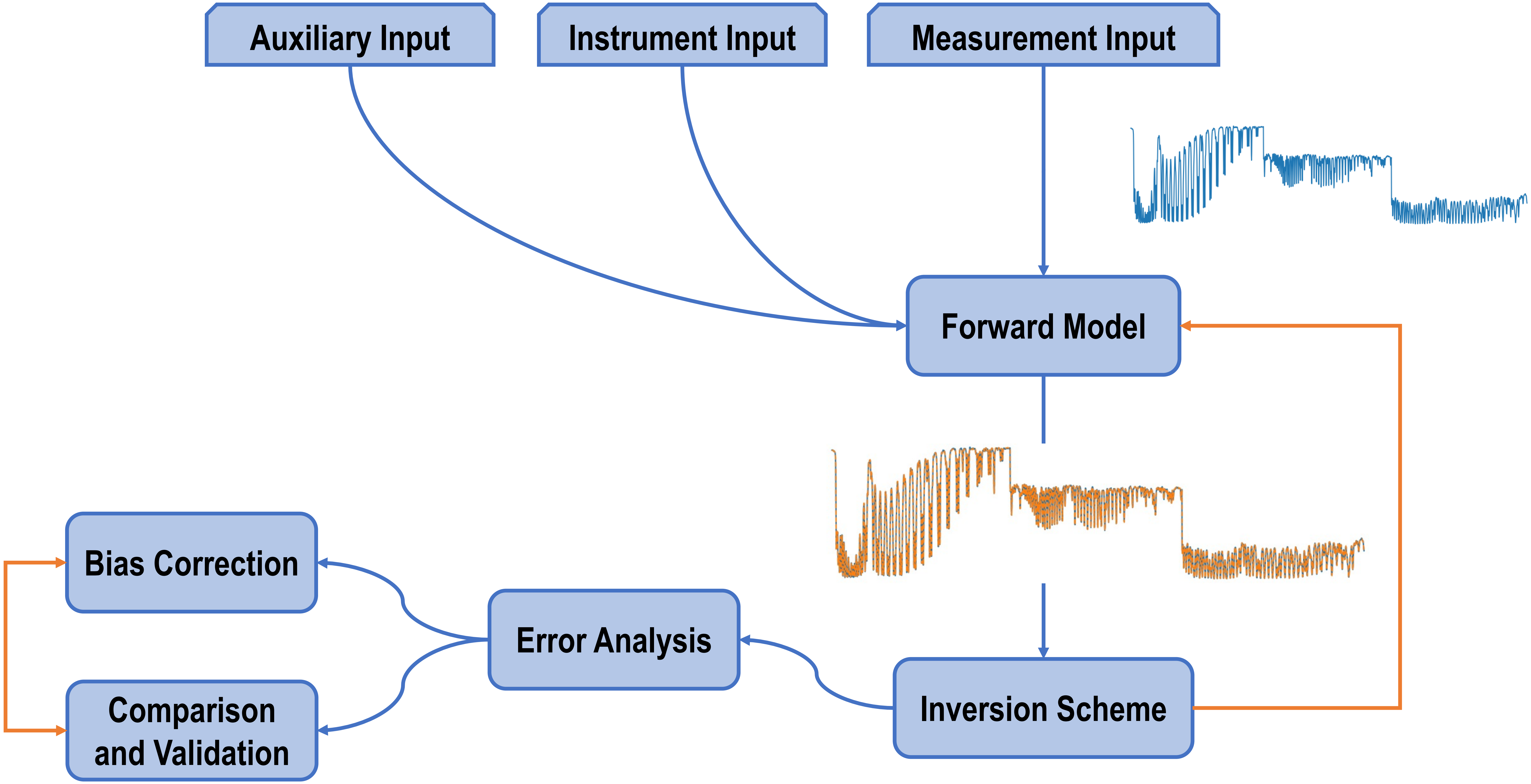
- **Accessible and open L2 algorithm toolkit**

Talk by Lesley Ott –
Today 3:30 PM!

Talk by Argyro Kavvada –
Friday at 9:00 AM!

Talk by Kevin Bowman –
Friday at 11:45 AM!

Identifying Needs in Algorithm Development



A Colorful Zoo of GHG L2 Algorithms

National Institute for Environmental Studies, Japan

Retrieval algorithm for CO_2 and CH_4 column abundances from short-wavelength infrared spectral observations by the Greenhouse gases observing satellite

Y. Yoshida[1], Y. Ota[1,*], N. Eguchi[1,**], N. Kikuchi[2], K. Nobuta[2], H. Tran[3], I. Morino[1], and T. Yokota[1]

University of Leicester

Atmospheric carbon dioxide retrieved from the Greenhouse gases Observing SATellite (GOSAT): Comparison with ground-based TCCON observations and GEOS-Chem model calculations

A. J. Cogan,[1] H. Boesch,[1] R. J. Parker,[1] L. Feng,[2] P. I. Palmer,[2] J.-F. L. Blavier,[3] N. M. Deutscher,[4] R. Macatangay,[5] J. Notholt,[4] C. Roehl,[3] T. Warneke,[4] and D. Wunch[3]

Universität Bremen

A Fast Atmospheric Trace Gas Retrieval for Hyperspectral Instruments Approximating Multiple Scattering—Part 1: Radiative Transfer and a Potential OCO-2 XCO_2 Retrieval Setup

Maximilian Reuter *, Michael Buchwitz, Oliver Schneising, Stefan Noël, Vladimir Rozanov, Heinrich Bovensmann and John P. Burrows

SRON Netherlands Institute for Space Research

Toward accurate CO_2 and CH_4 observations from GOSAT

A. Butz,[1,2] S. Guerlet,[2] O. Hasekamp,[2] D. Schepers,[2] A. Galli,[2] I. Aben,[2] C. Frankenberg,[3] J.-M. Hartmann,[4] H. Tran,[4] A. Kuze,[5] G. Keppel-Aleks,[6] G. Toon,[3] D. Wunch,[6] P. Wennberg,[6] N. Deutscher,[7,8] D. Griffith,[7] R. Macatangay,[7] J. Messerschmidt,[8] J. Notholt,[8] and T. Warneke[8]

Colorado State University — Jet Propulsion Laboratory, California Institute of Technology

The ACOS CO_2 retrieval algorithm – Part 1: Description and validation against synthetic observations

C. W. O'Dell[1], B. Connor[2], H. Bösch[3], D. O'Brien[1], C. Frankenberg[4], R. Castano[4], M. Christi[1], D. Eldering[4], B. Fisher[4], M. Gunson[4], J. McDuffie[4], C. E. Miller[4], V. Natraj[4], F. Oyafuso[4], I. Polonsky[1], M. Smyth[4], T. Taylor[1], G. C. Toon[4], P. O. Wennberg[5], and D. Wunch[5]

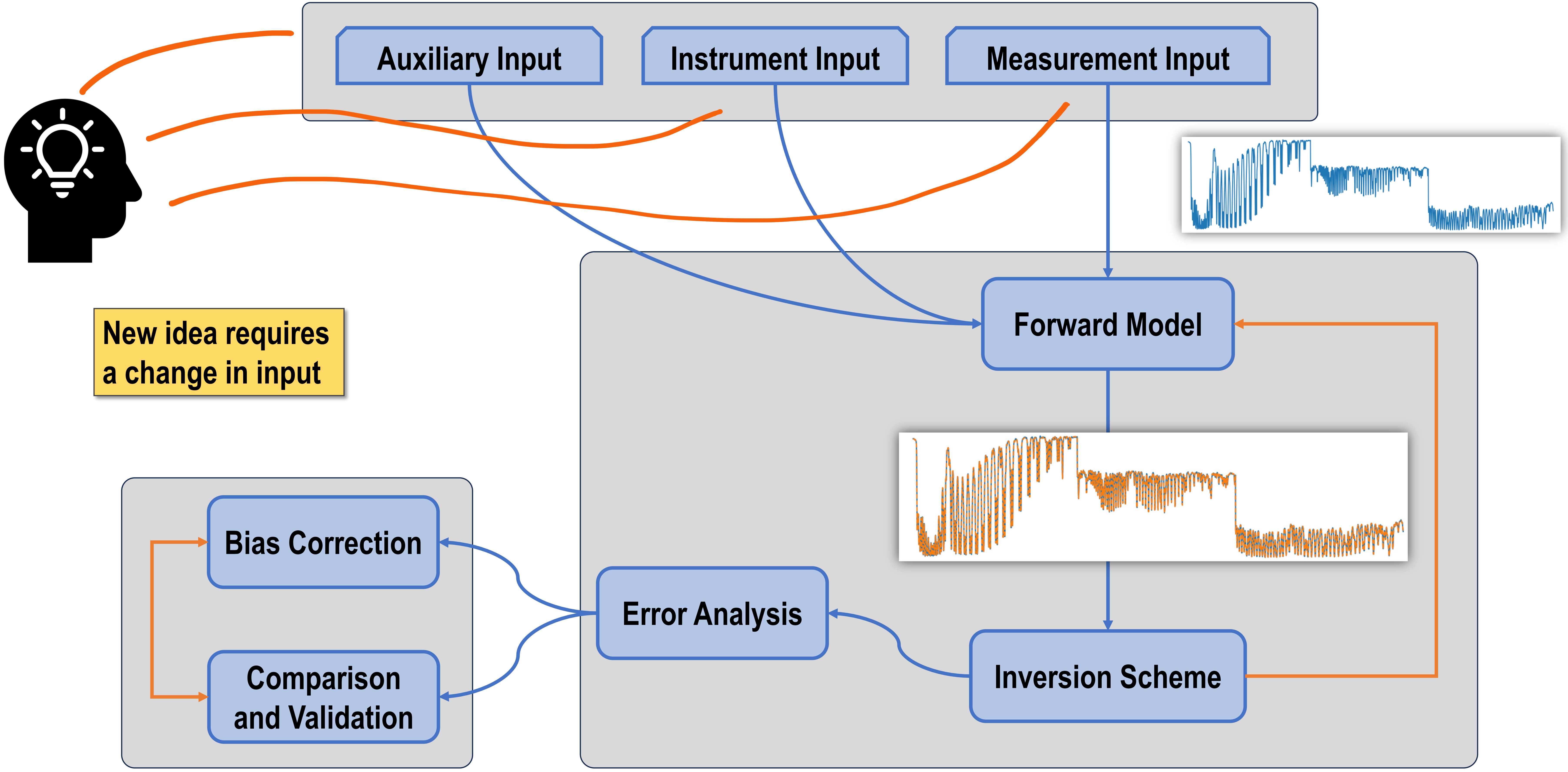
Yonsei University

Impact of Aerosol Property on the Accuracy of a CO_2 Retrieval Algorithm from Satellite Remote Sensing

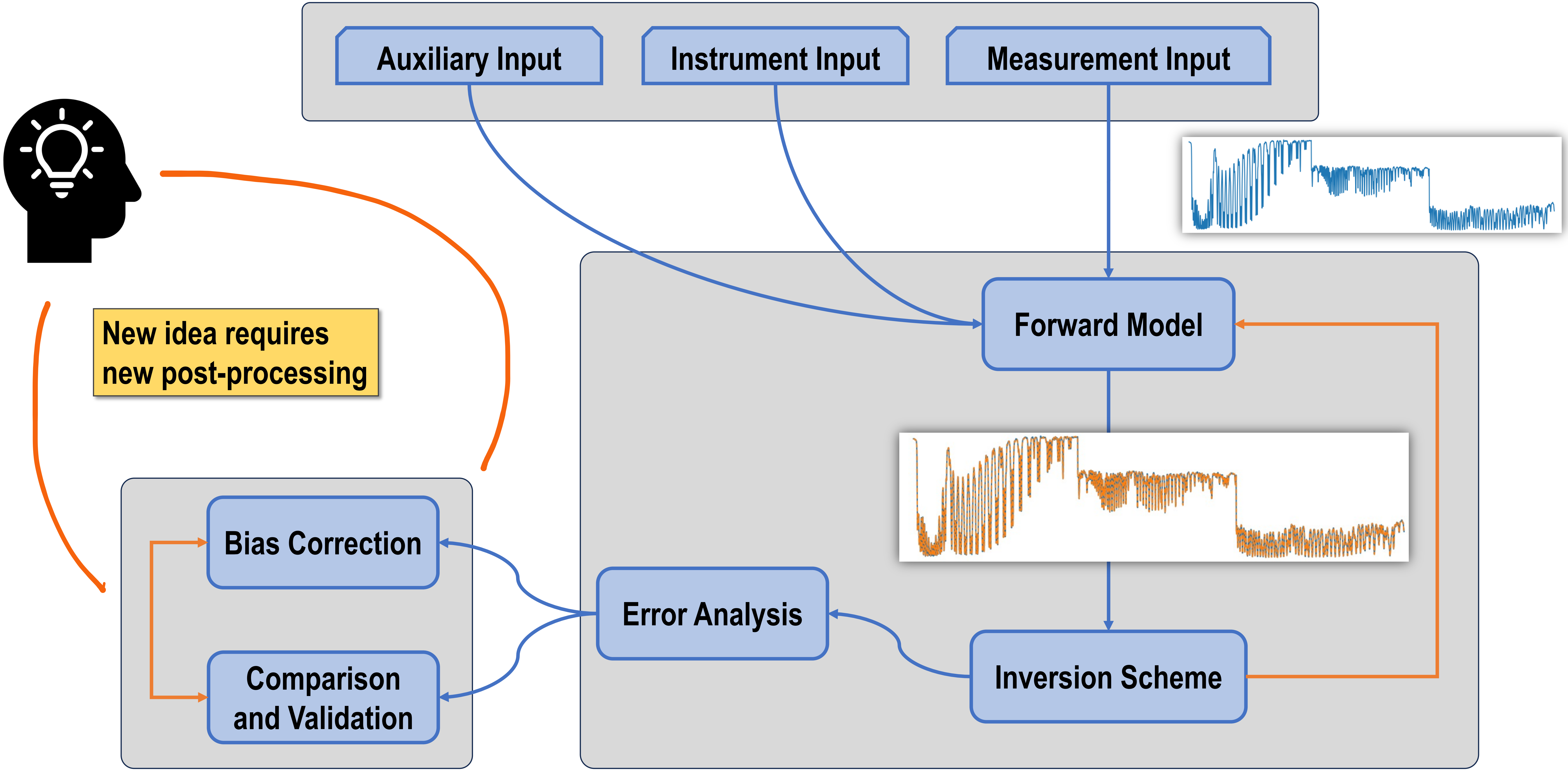
Yeonjin Jung [1], Jhoon Kim [1,*], Woogyung Kim [1], Hartmut Boesch [2,3], Hanlim Lee [4], Chunho Cho [5] and Tae-Young Goo [5]

and others!

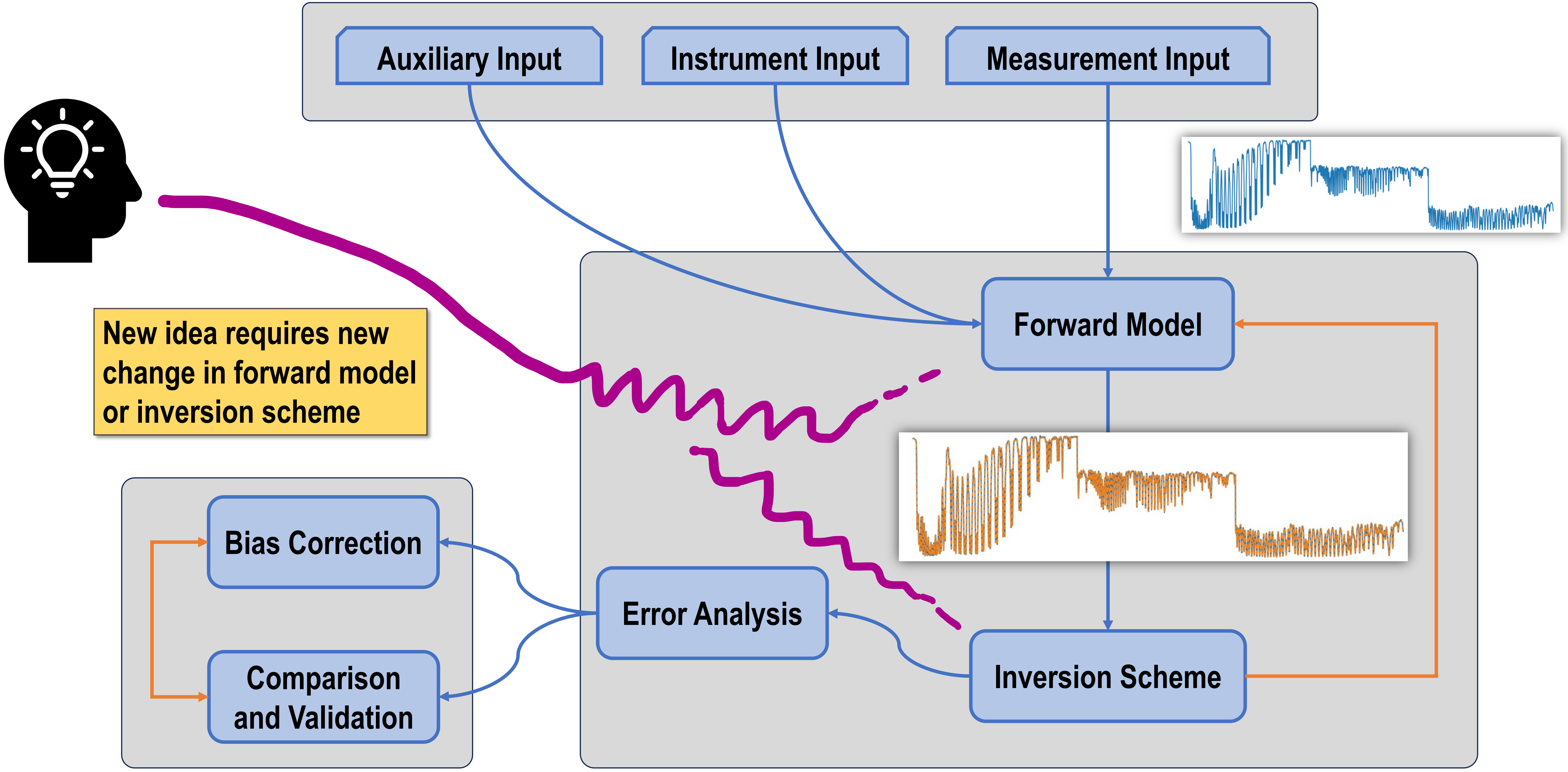
Identifying Needs in Algorithm Development



Identifying Needs in Algorithm Development



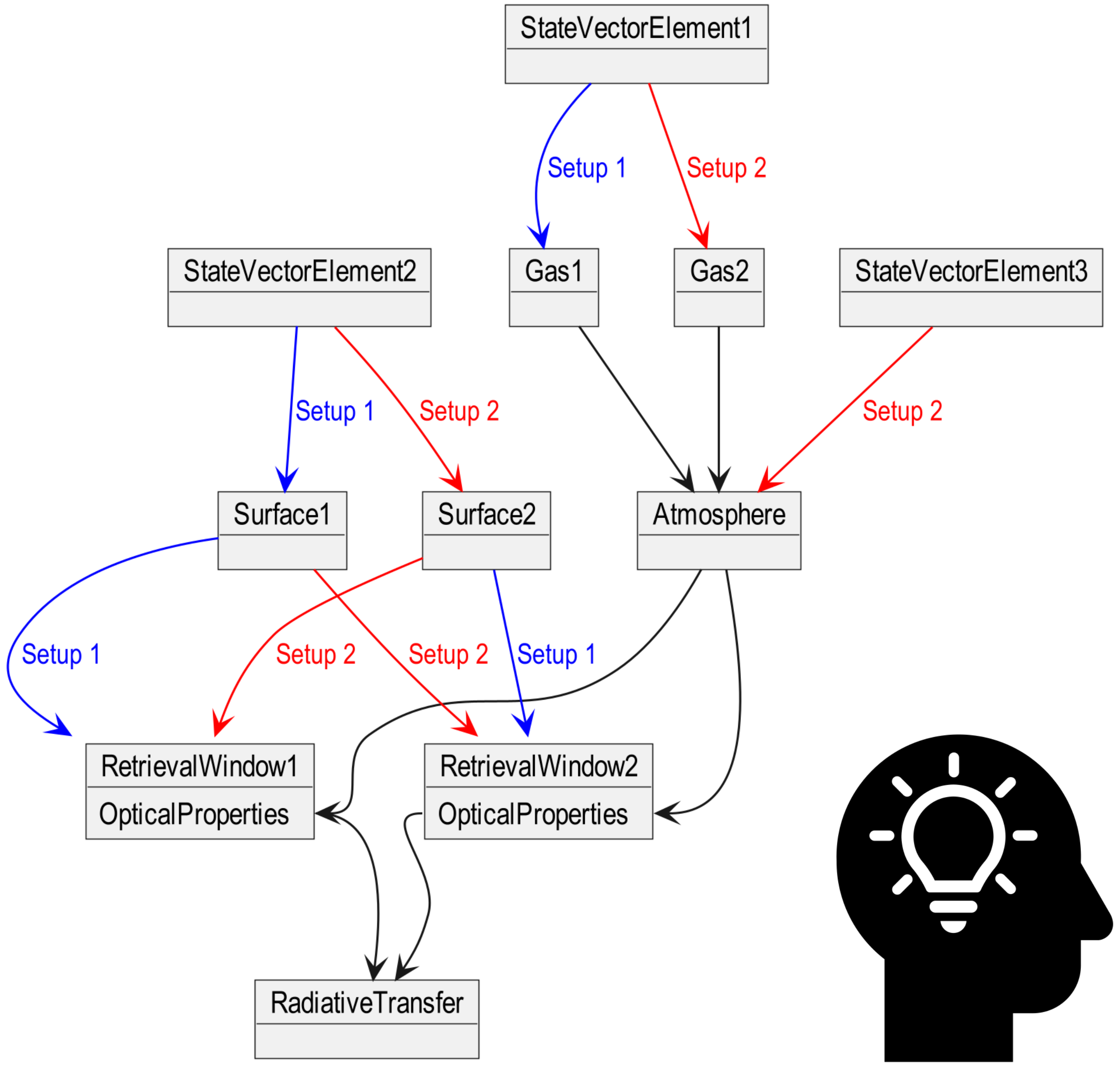
Identifying Needs in Algorithm Development

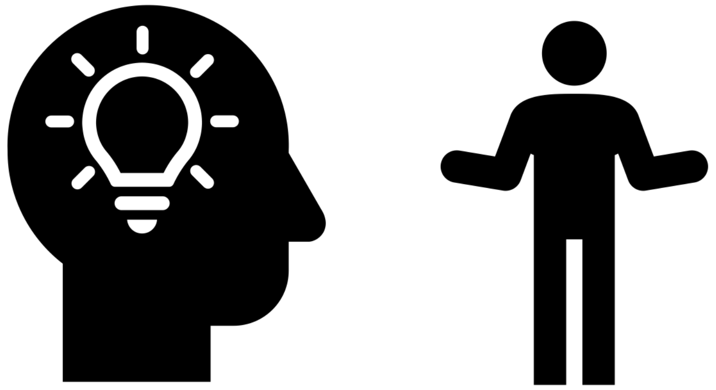


We Want Choices – Why Are Algorithms Complicated?

- GHG retrieval algorithms can be complex

- Many moving parts, a lot of cross-talk depending on the setup





Providing New, Accessible Tools

- Step 1

We need a well-performing program code that allows scientists to make meaningful adaptations to a retrieval algorithm

We keep a notion of "sequential code"

```
# Grab inverse of prior covariance needed for K^T Se^-1 K + Sa^-1
Sa_inv = inv(solver.prior_covariance)
# Instrument noise covariance
Se = create_Se_from_solver(solver)
Se_inv = inv(Se)
# Create jacobian matrix
K = create_K_from_solver(solver)
# Calculate inverse of posterior covariance matrix
Shat_inv = Sa_inv + K' * (Se_inv * K)
Shat = inv(Shat_inv)
# Calculate gain matrix
G = Shat * K' * Se_inv
# Calculate averaging kernel matrix
AK = G * K
```

julia

```
BenchmarkTools.Trial: 320 samples with 1 evaluation.
 Range (min … max):  349.056 ms … 544.585 ms  ┊ GC (min … max): 0.00% … 13.56%
 Time  (median):     365.958 ms              ┊ GC (median):    0.00%
 Time  (mean ± σ):   375.709 ms ±  24.240 ms ┊ GC (mean ± σ):  3.12% ±  5.44%

 349 ms           Histogram: frequency by time          431 ms <

 Memory estimate: 169.31 MiB, allocs estimate: 291975.
```

Single-iteration performance of a simplified 3-band OCO-type retrieval without scattering

Notebook-based workflows

Providing New, Accessible Tools

- Step 2

Documenting the code and its usage

How do I use function X?

What are common pitfalls?

Main

Design

Core concepts

Functions

- State Vector Functions

Types

- Buffer Types
- State Vector Types
  - Retrieval State Vector
  - State Vector Element (SVE) Types

G3RT does not provide a forward model per se, but rather provides the many components that users can utilize to build one, there are many functions which require a state vector type, as listed further in State Vector Functions.

Retrieval State Vector

A `RetrievalStateVector` can be instantiated by passing a vector of `AbstractStateVectorElement` variables.

`G3RT.RetrievalStateVector` — Type

A type to hold a collection of state vector elements for use in a retrieval.

- `state_vector_elements::Vector{G3RT.AbstractStateVectorElement}`

State Vector Element (SVE) Types

An `AbstractStateVector` consists of a list of state vector element (SVE) types, which are all subtypes of `AbstractStateVectorElement`. Every `AbstractStateVectorElement` type must be implemented as a `mutable struct`, due to the overall design of G3RT. Since instantiating new state vectors and state vector elements for each new retrieval scene would impact highly negatively on the overall performance, mutable structs are used. Once instantiated, users must make sure to reset each state vector element at the beginning of a new retrieval by setting the appropriate values and emptying out the `iterations` vector field with `empty!(sve.iterations)`, with `sve` being the state vector element variable.

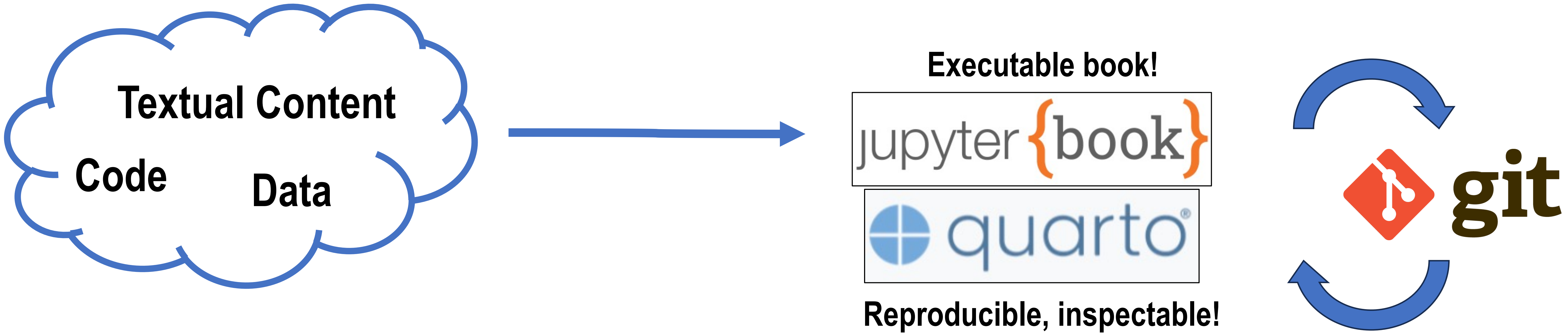
Why even use function Y?
How to do it efficiently?

Providing New, Accessible Tools

- Step 3

Documenting the underlying theory

Important: no complete forward model is provided, users must create their own but will retain ownership of their algorithm!



- Step 4

Providing examples to get you started!

Summary

- We are developing new, **accessible**, **performant**, modern L2 retrieval **algorithm tools** (tailored to GHGs, but with potential for use on other gases) that will be made available to the public

- These tools do not comprise a fully working retrieval set-up, users still have to connect all required parts, implement their own procedure – but therefore retain authorship, "the algorithm is **yours**"

- They will be shipped with documentation of the code, an ATBD referencing code sections, an **example implementations** that take the place of tutorials

Why all this? Our Motivations

- Support in-house development work
- Increase engagement with research groups that do not have a long history with GHG L2 work
  - Education & Training
- Make methods & data generation more transparent and accessible
- Support new ideas & science
- Opportunity for international collaborations

Thank you!

ReFRACtor (NASA/JPL)
(Python-interfaced framework on top of ACOS core modules)
https://github.com/ReFRACtor

Get in touch!
peter.somkuti@nasa.gov