Data Assimilation Algorithms

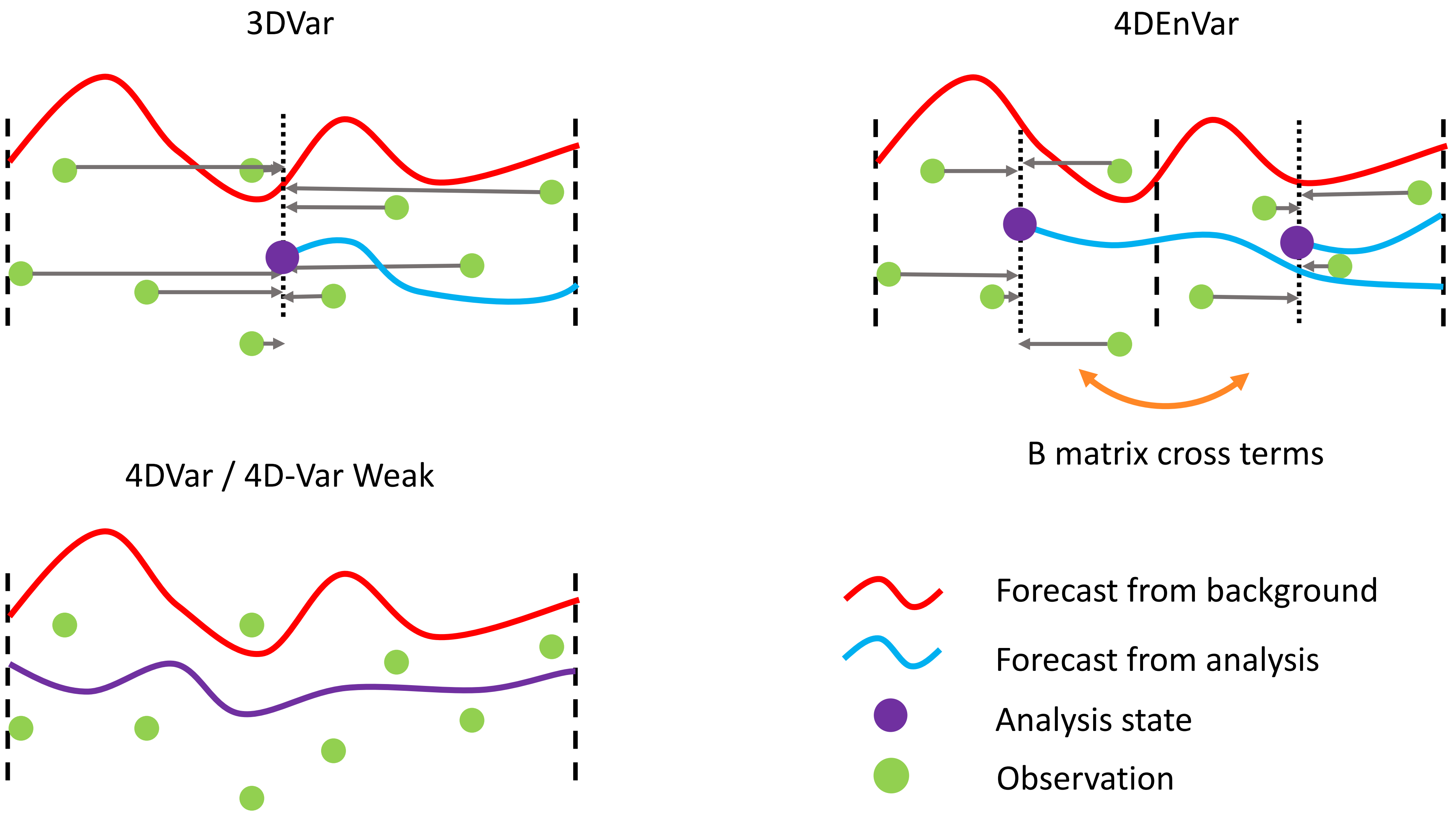
AMS Jedi Short Course

Introduction

JEDI supports several data assimilation algorithms working in multiple different modes.

- 3DVar, LETKF, 4DVar, 4DEnVar (parallel) and weak constraint 4DVar.
- Static B, Ensemble B and hybrid B using generic or specific implementations.
- Model space (primal) minimizers, observation space (dual) minimizers as well as double primal and saddle point minimizers. Several algorithms available.
- Ensemble based algorithms.

Differences between cost functions



General Variational Setup

```yaml
cost function:
  cost type: $(cost_function)
  window begin: '{{window_begin}}'
  window length: $(window_length)
  geometry:
    $(GEOMETRY)
  model:
    $(MODEL)
  analysis variables: $(an_variables)
  background:
    $(BACKGROUND)
  background error:
    $(BACKGROUND_ERROR)
  observations:
    $(OBSERVATIONS)
  constraints:
  - $(JC)
```

…

```yaml
variational:
  minimizer:
    $(MINIMIZER)
  iterations:
  - ninner: $(ninner)
    gradient norm reduction: $(reduc)
    geometry:
      $(GEOMETRY)
    linear model:
      $(LINEAR_MODEL)
    diagnostics:
      departures: $(diag)
final:
  diagnostics:
    departures: oman
  prints:
    frequency: PT3H
output:
  $(AN_OUTPUT)
```

$(cost_function) can be

- 3D-Var
- 4D-Var
- 4D-Ens-Var
- 4D-Weak

3D-Var

```yaml
cost function:
  cost type: $(cost_function)
  window begin: '{{window_begin}}'
  window length: $(window_length)
  geometry:
    $(GEOMETRY)
  model:
    $(MODEL)
  analysis variables: $(an_variables)
  background:
    $(BACKGROUND)
  background error:
    $(BACKGROUND_ERROR)
  observations:
    $(OBSERVATIONS)
  constraints:
  - $(JC)
```

```yaml
variational:
  minimizer:
    $(MINIMIZER)
  iterations:
  - ninner: $(ninner)
    gradient norm reduction: $(reduc)
    geometry:
      $(GEOMETRY)
    linear model:
      $(LINEAR_MODEL)
    diagnostics:
      departures: $(diag)
final:
  diagnostics:
    departures: oman
  prints:
    frequency: PT3H
output:
  $(AN_OUTPUT)
```

Model and LinearModel are not needed for 3DVar.

FGAT

First Guess and Appropriate Time (FGAT) is a special case of the 4DVar cost function, where the generic identity linear model is used in place of the actual LinearModel

```
cost function:
  cost type: $(cost_function)
  window begin: '{{window_begin}}'
  window length: $(window_length)
  geometry:
    $(GEOMETRY)
  model:
    $(MODEL)
  analysis variables: $(an_variables)
  background:
    $(BACKGROUND)
  background error:
    $(BACKGROUND_ERROR)
  observations:
    $(OBSERVATIONS)
  constraints:
  - $(JC)
```

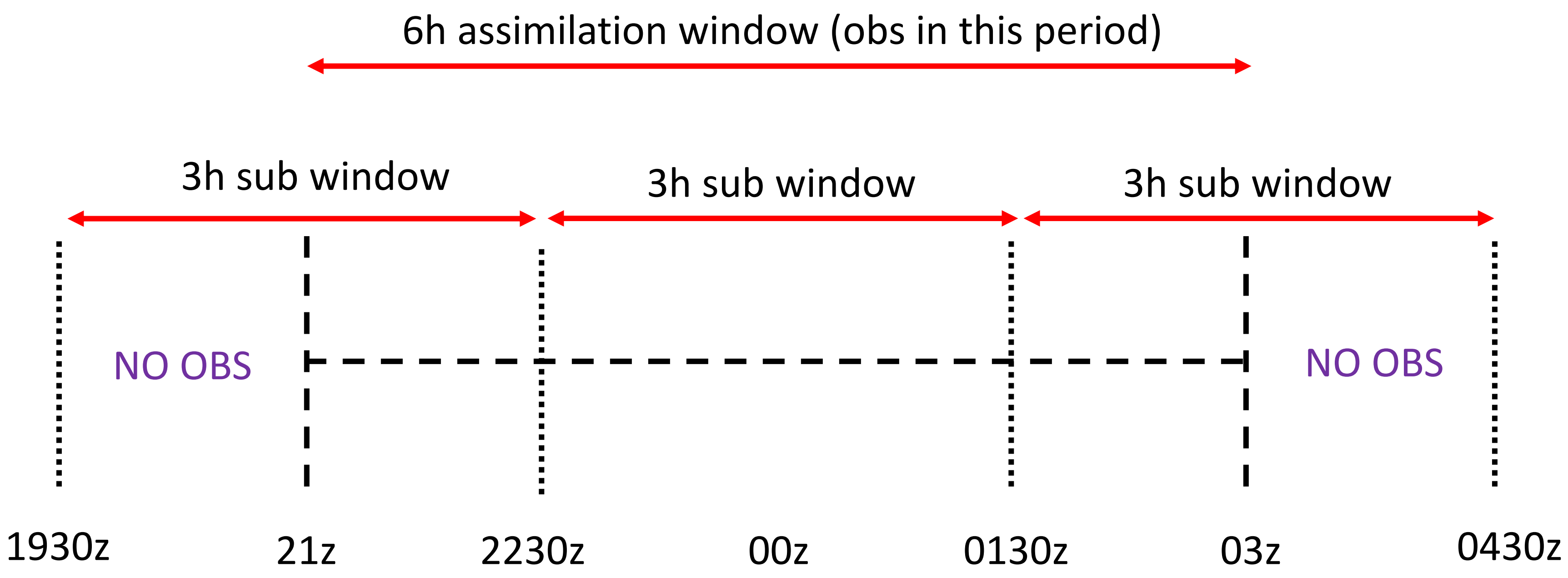
```
variational:
  minimizer:
    $(MINIMIZER)
  iterations:
  - ninner: $(ninner)
    gradient norm reduction: $(reduc)
    geometry:
      $(GEOMETRY)
    linear model:
      $(LINEAR_MODEL)
    diagnostics:
      departures: $(diag)
final:
  diagnostics:
    departures: oman
  prints:
    frequency: PT3H
output:
  $(AN_OUTPUT)
```

```
linear model:
  variable change: Identity
  tstep: PT3H
  tlm variables: *vars
```

4D-Ens-Var

```
cost function:
  cost type: 4D-Ens-Var
  analysis variables: [ua,va,T,ps,sphum,ice_wat,liq_wat,o3mr]
  window begin: '2018-04-14T21:00:00Z'
  window length: PT6H
  subwindow: PT3H
```

```
background:
  states:
  - $(state1) #21z
  - $(state2) #00z
  - $(state3) #03z
```



LETKF

```yaml
window begin: '{{window_begin}}'
window length: $(window_length)
geometry:
  $(GEOMETRY)
background:
 date: '{{window_begin}}'
 members:
   - $(MEMBER1)
   - $(MEMBER2)
   - ...
   - $(MEMBERN)
observations:
  $(OBSERVATIONS)
prints:
  frequency: PT3H
driver:
local ensemble DA:
  solver: LETKF
  inflation:
    rtps: $(rtps)
    rtpp: $(rtpp)
    mult: $(mult)
output:
  $(AN_OUTPUT)
```

LETKF is not part of the variational data assimilation algorithm and is a separate application.

The inputs are quite a bit simpler than variational, not requiring the model or background error and instead the ensemble states.

The user needs to specify some inflation and localization length scales to be applied in the on-the-fly background error modeling.

Background Error

```yaml
background error:
  covariance model: hybrid
  components:
  - covariance:
      covariance model: $(STATIC)
      date: {{window_begin}}
    weight:
      value: $(STATIC_WEIGHT)
  - covariance:
      covariance model: ensemble
      members:
        - $(ENS1)
        - $(ENS1)
        - ...
        - $(ENSN)
      localization:
        localization variables: *vars
        localization method: $(LOCALIZATION)
```

Background error model can be hybrid, static or ensemble. Localization is optional.

You can chain together as many B models as needed.

4D-Ens-Var

4D-Ens-Var requires ensemble states at the middle of each sub window:

```yaml
- states:
  - $(ens1_time1)
  - $(ens1_time2)
  - $(ens1_time3)
- states:
  - $(ens2_time1)
  - $(ens2_time2)
  - $(ens2_time3)
- ...
```

Minimizer

```
variational:
  minimizer:
    $(MINIMIZER)
  iterations:
  - ninner: $(ninner)
    gradient norm reduction: $(reduc)
    geometry:
      $(GEOMETRY)
    linear model:
      $(LINEAR_MODEL)
    diagnostics:
      departures: $(diag)
```

The variational part of the configuration controls the minimization of the cost function.

- DRGMRESR
- DRIPCG
- GMRESR
- IPCG
- SaddlePoint
- RPCG
- DRPCG
- DRPFOM
- LBGMRESR
- DRPLanczos
- PCG
- PLanczos
- RPLanczos
- MINRES
- FGMRES

$(ninner) controls the number of steps in the minimization.

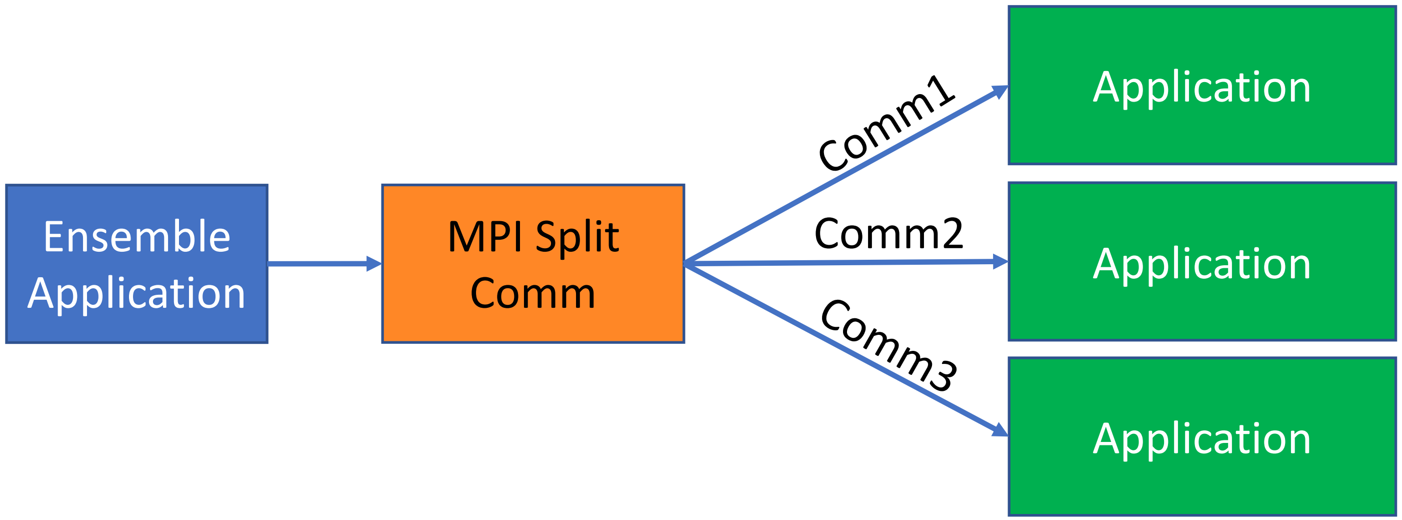
$(reduc) is the gradient norm reduction value for which the system is converged.

$(diag) is the diagnostics to produce from the system, e.g. omb.

Ensemble Applications

Any JEDI application can be run in groups and therefore used for ensemble applications such as ensemble forecast, ensemble H(x), ensemble data assimilation.

You pass the EnsembleApplication a list of yaml files and it splits the jobs.



eda.yaml

```yaml
files:
- "testinput/eda_3dvar_1.yaml"
- "testinput/eda_3dvar_2.yaml"
- "testinput/eda_3dvar_3.yaml"
```

ensemble_forecast.yaml

```yaml
files:
  - "testinput/ens_forecast_1.yaml"
  - "testinput/ens_forecast_2.yaml"
```

JEDI: Applications

All the Applications that are available can be found in `src/oops/runs`:

- `AddIncrement.h`
- `ConvertState.h`
- `DiffStates.h`
- `Dirac.h`
- `EnsRecenter.h`
- `EnsVariance.h`
- `ExternalDFI.h`
- `Forecast.h`
- `GenEnsPertB.h`
- `HofX.h`
- `HofXNoModel.h`
- `LocalEnsembleDA.h`
- `RTPP.h`
- `StaticBInit.h`
- `Variational.h`

Questions?