Isca Guide

Neil Lewis

June 24, 2018

Contents

1 Introduction

'Isca' is a framework for the idealised three-dimensional modelling of planetary atmospheres. From Isca, one can derive general circulation models (GCMs) of varying complexity.

In its most idealised form, Isca can be configured as a 'dry model' where water vapour is not represented in the model atmosphere, and thermal forcing is applied via Newtonian thermal relaxation.

Isca can also be configured as a 'moist model'. In this case Isca solves the primitive equations for a moist atmosphere where water vapour is represented and advected with the flow, and its interactions with the atmosphere and surface by means of latent heating effects are accounted for. Evaporation of water vapour occurs at the surface, and the 'fast' condensation of water vapour, whereby water vapour is removed as precipitation that reaches the surface immediately, occurs aloft. This treatment of condensation means that

there is no explicit liquid water content in the model atmosphere, and thus by extension, no representation of clouds.

The complexity of the moist model can itself be varied by choosing from a range of different parametrisations (i.e. statistical representations) for different processes. In the case of a dry model, as an example, one can increase the complexity of the equilibrium temperature towards which the temperature field is relaxed. For the moist model, different parametrisations for convection, radiation and surface processes are available. For a description of the available options, the reader is referred to Vallis et al. (2018).

2 Running Isca

Isca is easily configured and run using Python scripts and a python module called `isca`. The underlying model itself is written in Fortran. This section provides instructions on how to use the `isca` module to configure and run your own Isca experiments!

A few example Isca configurations, including the two that will be introduced in this section, can be found in the 'Isca' directory, at: `Isca/exp/test_cases/`.

This section will introduce two simple configurations for Isca:

1. A dry model, configured following Held and Suarez (1994). In this model, there is no representation of water vapour. Additionally thermal forcing is applied in the form of a Newtonian thermal relaxation, in place of explicitly considering radiative transfer.
2. A simple moist model, configured following Frierson et al. (2006). In this model, water vapour is represented and advected with the flow. Thermal forcing is applied by integrating the two-stream radiative transfer equations to calculate longwave and shortwave radiative fluxes.

2.1 Using a Unix terminal

As a brief preliminary, this subsection provides instructions for how to use and navigate around a Unix terminal.

Suppose you have just logged into a Unix machine (perhaps by using `ssh`). You will currently be 'located' in your home directory (or folder). To find out where you are 'located', use the `pwd` command ('print working directory'; type `pwd` and press `return`).

To see what is in a directory, say the one your are currently in, use the command `ls` ('list segments').

Suppose you wanted to get into your `Desktop` folder (if it exists, use `ls` to find out), you would use the command `cd` ('change directory') as follows:

```
cd Desktop/
```

Suppose you wanted to get back, you would enter:

```
cd ../  or  cd ~/
```

where the `cd ../` command just takes you back *one* directory, but the `cd ~/` command will always take you to your home directory.

Finally, to make a new directory you can use the `mkdir` ('make directory') command, for example:

```
mkdir test_directory
```

If you use `ls` it will now appear, and you could enter it with `cd`. To remove it you would use: `rm -r test_directory`, where `rm` means remove, and `-r` means recursive, as this command would also remove all files within the directory. To remove an individual file you would use `rm file_name.extension`.

2.2 A simple dry model experiment

2.2.1 Background

We will first run an experiment where Isca is configured as a 'dry' GCM, where there is no representation of water vapour, and additionally thermal forcing is applied in the form of a Newtonian thermal relaxation, in place of explicitly considering radiative transfer. The configuration here follows Held and Suarez (1994).

Thermal forcing is applied by 'relaxing' temperature to a prescribed equilibrium temperature profile:

$$\frac{\partial T}{\partial t} = \cdots - k_{\mathrm{T}}(\phi, \sigma)\left[T - T_{\mathrm{eq}}(\phi, \sigma)\right], \tag{1}$$

where T is the kinematic temperature, k_{T} is the coefficient of thermal relaxation, and T_{eq} is the thermal relaxation profile. ϕ is latitude, and $\sigma = p/p_{\mathrm{s}}$ is pressure normalised by the surface pressure.

2.2.2 Configuring the experiment

The Held-Suarez test experiment is one of the example configurations that is supplied with Isca. The python script that configures it is called '`held_suarez_test_case.py`' and can be found at `Isca/exp/test_cases/held_suarez/`. Navigate to it using, `cd`, and open it with:
'`emacs held_suarez_test_case.py &`' (to check if you are in the right place, you can use `pwd` and `ls`).

A window should now open containing the code presented in Figure 1.

We will briefly now walk through the main commands used in the python script:

- The '`from isca import ...`' line imports the `isca` python module we installed earlier. This tells python to load the tools we will need to configure and run Isca. Note that here we import '`DryCodeBase`'. As here we are only intending on running a dry model, we only import the code necessary for this.
- `NCORES` specifies how many cores should be used when running the model.
- `RESOLUTION` this specifies the horizontal (in this case, spectral `T42`) and vertical (`25` pressure levels) resolution at which the model should run.
- `cb = DryCodeBase.from_directory(GFDL_BASE)` tells Isca to find the model code at the location `GFDL_BASE` that we specified earlier.

```python
import numpy as np

from isca import DryCodeBase, DiagTable, Experiment, Namelist, GFDL_BASE

NCORES = 16
RESOLUTION = 'T42', 25  # T42 horizontal resolution, 25 levels in pressure

# a CodeBase can be a directory on the computer,
# useful for iterative development
cb = DryCodeBase.from_directory(GFDL_BASE)

# or it can point to a specific git repo and commit id.
# This method should ensure future, independent, reproducibility of results.
# cb = DryCodeBase.from_repo(repo='https://github.com/isca/isca', commit='isca1.1')

# compilation depends on computer specific settings.  The $GFDL_ENV
# environment variable is used to determine which `$GFDL_BASE/src/extra/env` file
# is used to load the correct compilers.  The env file is always loaded from
# $GFDL_BASE and not the checked out git repo.

cb.compile()  # compile the source code to working directory $GFDL_WORK/codebase

# create an Experiment object to handle the configuration of model parameters
# and output diagnostics

exp_name = 'held_suarez_default'
exp = Experiment(exp_name, codebase=cb)

#Tell model how to write diagnostics
diag = DiagTable()
diag.add_file('atmos_monthly', 1, 'days', time_units='days')

#Tell model which diagnostics to write
diag.add_field('dynamics', 'ps', time_avg=True)
diag.add_field('dynamics', 'bk')
diag.add_field('dynamics', 'pk')
diag.add_field('dynamics', 'ucomp', time_avg=True)
diag.add_field('dynamics', 'vcomp', time_avg=True)
diag.add_field('dynamics', 'temp', time_avg=True)
diag.add_field('dynamics', 'vor', time_avg=True)
diag.add_field('dynamics', 'div', time_avg=True)

exp.diag_table = diag

# define namelist values as python dictionary
# wrapped as a namelist object.
namelist = Namelist({
    'main_nml': {
        'dt_atmos': 600,
        'days': 1,
        'calendar': 'thirty_day',
        'current_date': [2000,1,1,0,0,0]
    },

    'atmosphere_nml': {
        'idealized_moist_model': False  # False for Newtonian Cooling.  True for Isca/Frierson
    },

    'spectral_dynamics_nml': {
        'damping_order'           : 4,                    # default: 2
        'water_correction_limit'  : 200.e2,               # default: 0
        'reference_sea_level_press': 1.0e5,               # default: 101325
        'valid_range_t'           : [100., 800.],         # default: (100, 500)
        'initial_sphum'           : 0.0,                  # default: 0
        'vert_coord_option'       : 'uneven_sigma',       # default: 'even_sigma'
        'scale_heights': 6.0,
        'exponent': 7.5,
        'surf_res': 0.5
    },

    # configure the relaxation profile
    'hs_forcing_nml': {
        't_zero': 315.,    # temperature at reference pressure at equator (default 315K)
        't_strat': 200.,   # stratosphere temperature (default 200K)
        'delh': 60.,       # equator-pole temp gradient (default 60K)
        'delv': 10.,       # lapse rate (default 10K)
        'eps': 0.,         # stratospheric latitudinal variation (default 0K)
        'sigma_b': 0.7,    # boundary layer friction height (default p/ps = sigma = 0.7)

        # negative sign is a flag indicating that the units are days
        'ka':   -40.,      # Constant Newtonian cooling timescale (default 40 days)
        'ks':    -4.,      # Boundary layer dependent cooling timescale (default 4 days)
        'kf':   -1.,       # BL momentum frictional timescale (default 1 days)

        'do_conserve_energy':   True,  # convert dissipated momentum into heat (default True)
    },

    'diag_manager_nml': {
        'mix_snapshot_average_fields': False
    },

    'fms_nml': {
        'domains_stack_size': 600000                        # default: 0
    },

    'fms_io_nml': {
        'threading_write': 'single',                         # default: multi
        'fileset_write': 'single',                           # default: multi
    }
})

exp.namelist = namelist
exp.set_resolution(*RESOLUTION)

#Lets do a run!
if __name__ == '__main__':
    exp.run(1, num_cores=NCORES, use_restart=False)
    for i in range(2, 13):
        exp.run(i, num_cores=NCORES)  # use the restart i-1 by default
```

Figure 1: Python script configuration for Held-Suarez test case.

- `cb.compile()` compiles the model code.
- The '`exp = Experiment(...)`' line creates an experiment object. This is what we will configure for our specific experiment.
- `diag = DiagTable()` creates a `DiagTable` object which we can configure to tell Isca which variables to output (e.g Temperature), and how they should be averaged. `diag.add_file('atmos_monthly',1,'days',time_units='days')` creates an output file, `atmos_monthly.nc` (can be found in `isca_data/exp_name/run#/`) which contains the output from the model run. For this specific example, output will be written to it every one day. The '`diag.add_field`' commands tell Isca which fields should be written to `atmos_monthly.nc`. A full list of possible variables to output can be found in `Isca/FieldNames.md` which can be opened using '`cd Isca; sublime FieldNames.md &`'.
- `exp.diag_table = diag` tells our experiment to use the DiagTable we have just created.
- `namelist = Namelist({...})` defines a 'namelist' object. It is within the namelist that we will configure all of the models 'science options'. To do this, a number of smaller 'component specific' namelists (end with `_nml`) are modified. All of the options changed when configuring the namelist are being changed from default parameters. It is therefore only necessary to set the values of particular namelist variables that we need to change from default. A few of the main options selected in `held_suarez_test_case.py` will now be mentioned:

  1. `main_nml` is most commonly altered to change the model run time properties. For example, `dt_atmos` is the model timestep (in seconds), and `months` specifies the number of months the model should run for. In this case the base experiment is initially configured to run for one day, and a timestep of 600 seconds. Suppose we wanted the model to run for 13 months and one day, we could write `'months': 13,'days': 1`.
  2. Where we set:

     `'atmosphere_nml':{'idealized_moist_model': False},`

     we are telling `isca` that we would like to run a dry model.
  3. `spectral_dynamics_nml` configures the dynamical core, and `hs_forcing_nml` configures the Newtonian cooling given by Equation 1.
- `exp.namelist = namelist` tells the experiment object to use the namelist we have just configured. `exp.set_resolution` tells it to use the resolution we defined earlier.
- Finally, we need to tell the model how we would like it to run. Each of the `exp.run` commands will ask Isca to run for the amount of time specified in `main_nml`. Note that the first `exp.run` command specifies `use_restart=False`. This tells Isca that for this first run, we would like it to 'start from scratch' (default is isothermal atmosphere). The subsequent `exp.run` calls will restart from where the previous left off. Each `exp.run` call will produce its own `atmos_monthly.nc` file. In the example given in Figure 1, the model is asked by `main_nml` to run for 1 month. Thus, the first `exp.run` command and the `exp.run` commands in the `for` loop will run this experiment for a total of 12[1] months. As in the `DiagTable` (again, in Figure 1) it has been specified that the model should output time averaged data every one month, this model will produce 12 `atmos_monthly.nc`

[1] `range(2,13)=[1,2,3,...,12]`

files: each containing time averaged data over one month. If we wanted all of the output to be in one `atmos_monthly.nc` file, then we could ask the model to run for one year in `main_nml`, but still ask the `DiagTable` for monthly data. To run the model for one year we would then write `exp.run` only once (i.e. delete the loop with subsequent calls).

In the next section, we will run the experiment!

2.2.3 Running the experiment

Running the experiment configured in the last section is as simple as running the python script `held_suarez_test_case.py`. To do this, you would type:

```
nice -10 python held_suarez_test_case.py,
```

where the nice command simply lowers the priority of the job, so more important jobs (e.g. system processes, or important jobs run by others) will take priority. *Note: To run on Isca (the University of Exeter supercomputer), or Argo, you will need to use a submission script similar to those created by Stephen Thompson for the ICTP summer school experiments.*

2.3 A simple moist model experiment

Background

We now consider a model where water vapour is included in the model atmosphere and advected with the flow. Hence, water vapour interacts with the dynamics via the latent heat release associated with evaporation and condensation. The set-up for this model follows Frierson et al. (2006).

Thermal forcing is applied by calculating longwave (thermal emission from the planet and atmosphere) and shortwave (radiation from the Sun) radiative fluxes. To calculate radiative fluxes, the model integrates the two-stream radiative transfer equations:

$$\frac{\mathrm{d}U}{\mathrm{d}\tau} = U - B, \tag{2}$$

$$\frac{\mathrm{d}D}{\mathrm{d}\tau} = B - D, \tag{3}$$

where U is the upward radiative flux, D is the downward flux, and $B = \sigma T^4$. τ is the optical depth, and describes how much radiation is absorbed by the atmosphere.

In this model, the atmosphere is assumed to be transparent to shortwave radiation. Thus shortwave radiation only interacts with the surface, where it is either reflected or absorbed, and $\tau_{\mathrm{sw}} = 0$ across all shortwave wavelengths. With Isca, it is easy to relax this assumption and instead define τ_{sw} to have a vertical profile that exponentially decays with height.

In the model considered here, the longwave radiative transfer makes the 'gray' atmosphere approximation. This means that a single absorption coefficient is used across all longwave wavelengths. In the longwave,

the optical depth is specified as follows:

$$\tau_{\mathrm{lw}} = \tau_0 \left[f_{\mathrm{l}} \left(\frac{p}{p_{\mathrm{s}}} \right) + (1 - f_{\mathrm{l}}) \left(\frac{p}{p_{\mathrm{s}}} \right)^4 \right], \tag{4}$$

where τ_0 is the surface optical depth (varies with latitude), and f_{l} is a parameter that determines the weighting of the two terms. The quartic term is chosen to approximate the structure of water vapour in the atmosphere, and the linear term is included to reduce stratospheric relaxation times. Note here that water vapour content (measured as specific humidity, q - the mass of water vapour per mass of air) is not used to calculate the longwave optical depth. The quartic pressure term instead approximates the distribution of water vapour, and changes in water vapour content do not affect the radiative transfer. If one wanted to include the effects of water vapour on the longwave radiative transfer, it is easy to change the prescription of τ to include a dependency on q. Options for doing this are easily selected with Isca.

2.3.1 Configuring the experiment

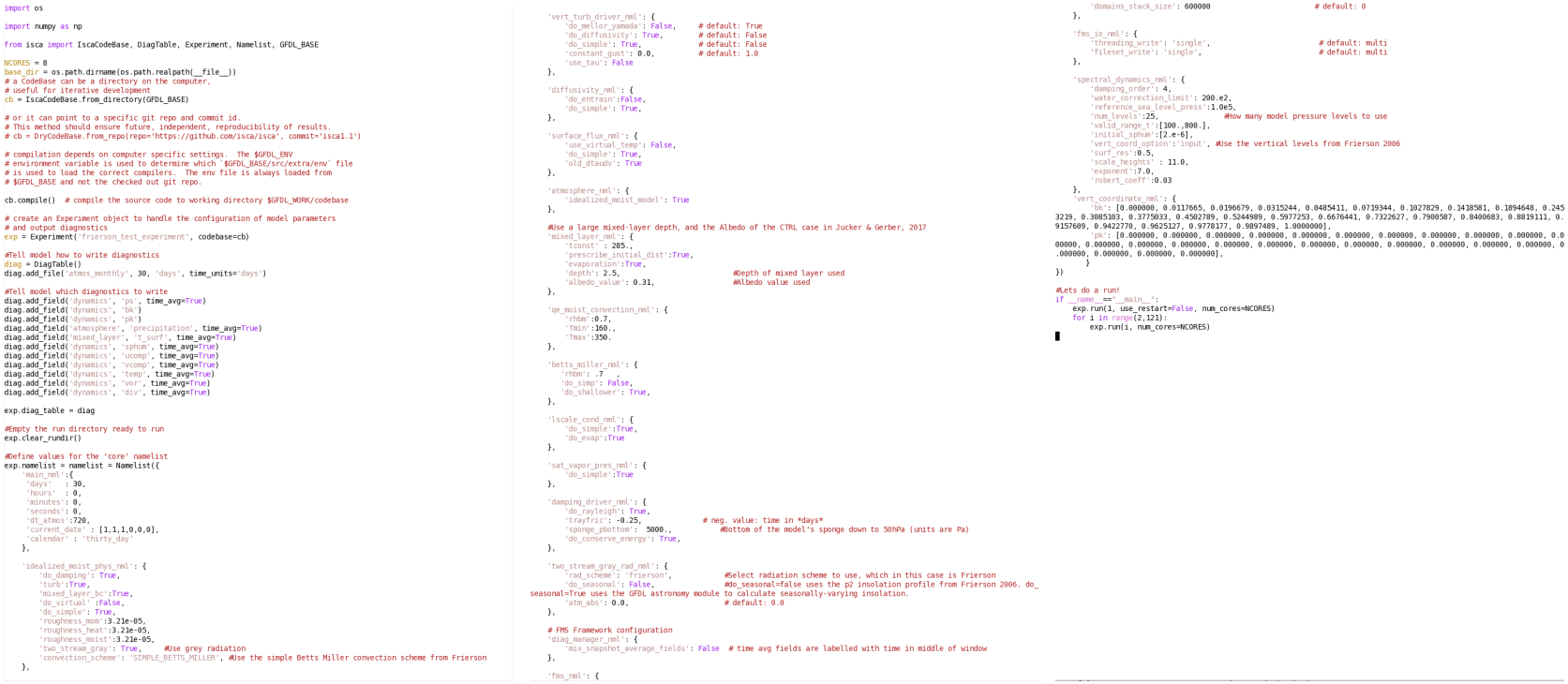
The Frierson test experiment is one of the example configurations that is supplied with Isca. The python script that configures it is called '`frierson_test_case.py`' and can be found at `Isca/exp/test_cases/frierson/`. Navigate to it using, `cd`, and open it with:
'`emacs frierson_test_case.py &`' (to check if you are in the right place, you can use `pwd` and `ls`).

A window should now open containing the code presented in Figure 2.

We will briefly now walk through the main commands used in the python script. This time, only options not present in the `held_suarez_test_case.py` configuration file will be discussed. See Section 2.2.2 for a full overview.

- This time, in the '`from isca import ...`' line, we import '`IscaCodeBase`'. We are intending on running a moist model, and so we import the code necessary for this.
- As before, `namelist = Namelist({...})` defines a 'namelist' object. Recall that it is within the namelist that we will configure all of the models 'science options'. A few of the main differences between options selected in `frierson_test_case.py` and `held_suarez_test_case.py` will now be mentioned:
  1. Where we set:
     `'atmosphere_nml':{'idealized_moist_model': True},`
     we are telling `isca` that we would like to run a moist model.
  2. The 'physics options', that is our choices for the parametrisation of convection, radiation, surface processes, etc are set in `'idealized_moist_phys_nml'`. Here `'two_stream_gray': True` tells the model to use some sort of gray radiation scheme, and `'convection_scheme': 'SIMPLE_BETTS_MILLER'` tells the model to use the quasi-equilibrium convection scheme of Frierson (2007). Each parametrisation is then configured with its own namelist:
  3. `'two_stream_gray_rad_nml'` configures the gray radiation scheme. Here we choose the `'frierson'` option to set the longwave optical depths to those described by Equation 4.
  4. `'qe_moist_convection_nml` configures the simple Betts-Miller convection scheme.

In the next section, we will run the experiment!



```
import os

import numpy as np

from isca import IscaCodeBase, DiagTable, Experiment, Namelist, GFDL_BASE

NCORES = 8
base_dir = os.path.dirname(os.path.realpath(__file__))
# a CodeBase can be a directory on the computer,
# useful for iterative development
cb = IscaCodeBase.from_directory(GFDL_BASE)

# or it can point to a specific git repo and commit id.
# This method should ensure future, independent, reproducibility of results.
# cb = DryCodeBase.from_repo(repo='https://github.com/isca/isca', commit='isca1.1')

# compilation depends on computer specific settings.  The $GFDL_ENV
# environment variable is used to determine which `$GFDL_BASE/src/extra/env` file
# is used to load the correct compilers.  The env file is always loaded from
# $GFDL_BASE and not the checked out git repo.

cb.compile()  # compile the source code to working directory $GFDL_WORK/codebase

# create an Experiment object to handle the configuration of model parameters
# and output diagnostics
exp = Experiment('frierson_test_experiment', codebase=cb)

#Tell model how to write diagnostics
diag = DiagTable()
diag.add_file('atmos_monthly', 30, 'days', time_units='days')

#Tell model which diagnostics to write
diag.add_field('dynamics', 'ps', time_avg=True)
diag.add_field('dynamics', 'bk')
diag.add_field('dynamics', 'pk')
diag.add_field('atmosphere', 'precipitation', time_avg=True)
diag.add_field('mixed_layer', 't_surf', time_avg=True)
diag.add_field('dynamics', 'sphum', time_avg=True)
diag.add_field('dynamics', 'ucomp', time_avg=True)
diag.add_field('dynamics', 'vcomp', time_avg=True)
diag.add_field('dynamics', 'temp', time_avg=True)
diag.add_field('dynamics', 'vor', time_avg=True)
diag.add_field('dynamics', 'div', time_avg=True)

exp.diag_table = diag

#Empty the run directory ready to run
exp.clear_rundir()

#Define values for the 'core' namelist
exp.namelist = namelist = Namelist({
    'main_nml':{
     'days'   : 30,
     'hours'  : 0,
     'minutes': 0,
     'seconds': 0,
     'dt_atmos':720,
     'current_date' : [1,1,1,0,0,0],
     'calendar' : 'thirty_day'
    },

    'idealized_moist_phys_nml': {
        'do_damping': True,
        'turb':True,
        'mixed_layer_bc':True,
        'do_virtual' :False,
        'do_simple': True,
        'roughness_mom':3.21e-05,
        'roughness_heat':3.21e-05,
        'roughness_moist':3.21e-05,
        'two_stream_gray': True,    #Use grey radiation
        'convection_scheme': 'SIMPLE_BETTS_MILLER', #Use the simple Betts Miller convection scheme from Frierson
    },

    'vert_turb_driver_nml': {
        'do_mellor_yamada': False,      # default: True
        'do_diffusivity': True,         # default: False
        'do_simple': True,              # default: False
        'constant_gust': 0.0,           # default: 1.0
        'use_tau': False
    },

    'diffusivity_nml': {
        'do_entrain':False,
        'do_simple': True,
    },

    'surface_flux_nml': {
        'use_virtual_temp': False,
        'do_simple': True,
        'old_dtaudv': True
    },

    'atmosphere_nml': {
        'idealized_moist_model': True
    },

    #Use a large mixed-layer depth, and the Albedo of the CTRL case in Jucker & Gerber, 2017
    'mixed_layer_nml': {
        'tconst' : 285.,
        'prescribe_initial_dist':True,
        'evaporation':True,
        'depth': 2.5,                   #Depth of mixed layer used
        'albedo_value': 0.31,           #Albedo value used
    },

    'qe_moist_convection_nml': {
        'rhbm':0.7,
        'Tmin':160.,
        'Tmax':350.
    },

    'betts_miller_nml': {
       'rhbm': .7   ,
       'do_simp': False,
       'do_shallower': True,
    },

    'lscale_cond_nml': {
        'do_simple':True,
        'do_evap':True
    },

    'sat_vapor_pres_nml': {
        'do_simple':True
    },

    'damping_driver_nml': {
        'do_rayleigh': True,
        'trayfric': -0.25,              # neg. value: time in *days*
        'sponge_pbottom':  5000.,       #Bottom of the model's sponge down to 50hPa (units are Pa)
        'do_conserve_energy': True,
    },

    'two_stream_gray_rad_nml': {
        'rad_scheme': 'frierson',       #Select radiation scheme to use, which in this case is Frierson
        'do_seasonal': False,           #do_seasonal=false uses the p2 insolation profile from Frierson 2006. do_
seasonal=True uses the GFDL astronomy module to calculate seasonally-varying insolation.
        'atm_abs': 0.0,                 # default: 0.0
    },

    # FMS Framework configuration
    'diag_manager_nml': {
        'mix_snapshot_average_fields': False  # time avg fields are labelled with time in middle of window
    },

    'fms_nml': {
        'domains_stack_size': 600000                        # default: 0
    },

    'fms_io_nml': {
        'threading_write': 'single',                         # default: multi
        'fileset_write': 'single',                           # default: multi
    },

    'spectral_dynamics_nml': {
        'damping_order': 4,
        'water_correction_limit': 200.e2,
        'reference_sea_level_press':1.0e5,
        'num_levels':25,                #How many model pressure levels to use
        'valid_range_t':[100.,800.],
        'initial_sphum':[2.e-6],
        'vert_coord_option':'input', #Use the vertical levels from Frierson 2006
        'surf_res':0.5,
        'scale_heights' : 11.0,
        'exponent':7.0,
        'robert_coeff':0.03
    },
    'vert_coordinate_nml': {
        'bk': [0.000000, 0.0117665, 0.0196679, 0.0315244, 0.0485411, 0.0719344, 0.1027829, 0.1418581, 0.1894648, 0.245
3219, 0.3085103, 0.3775033, 0.4502789, 0.5244989, 0.5977253, 0.6676441, 0.7322627, 0.7900587, 0.8400683, 0.8819111, 0.
9157609, 0.9422770, 0.9625127, 0.9778177, 0.9897489, 1.0000000],
        'pk': [0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.0
00000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0.000000, 0
.000000, 0.000000, 0.000000, 0.000000],
    }
})

#Lets do a run!
if __name__=='__main__':
    exp.run(1, use_restart=False, num_cores=NCORES)
    for i in range(2,121):
        exp.run(i, num_cores=NCORES)
```

Figure 2: Python script configuration for Frierson test case.

2.3.2 Running the experiment

Once again, the experiment can be run by running the python script, or in the case of a HPC, by using a submission script.

3 Designing your own experiments

By now, you should have some idea of how to configure Isca. How you choose to do so will depend on what you would like to investigate.

Some of the options available, that we have not discussed are:

- A radiative-convective equilibrium dry model.
- A suite of gray radiation schemes ranging from the Frierson scheme (introduced here, simplest), through gray schemes with a water vapour feedback, to a 'two-band' scheme with an infra-red window (most complex).
- Complex radiation schemes: RRTM and SOCRATES (Met Office radiation scheme).
- Convection schemes ranging from dry adjustment to mass-flux schemes (Relaxed Arakawa-Schubert).
- Idealised continents (i.e simple boxes and triangles) and realistic continents (the Frierson model, which we considered, is an aquaplanet model; it has no continents).
- Generic orbital parameters: one can change the planets rotation rate, obliquity, and orbital eccentricity.

Below, a mini 'reading' list is provided, which contains studies that have used idealised general circulation modelling to reveal a little more about the nature of atmospheric dynamics.

Studies using Isca:

1. Thomson and Vallis, 2018: Atmospheric Response to SST anomalies. Part 1: Background-state dependence, teleconnections and local effects in winter (Part 1) and in Summer (Part 2). *J. Atmos. Sci.,* submitted. (available at Geoff's website.)
2. Geen et al., 2018: Regime Change Behaviour During Asian Monsoon Onset. *J. Climate,* **31,** 3327-3348.
3. Vallis et al., 2018: Isca, v1.0: A framework for the global modelling of the atmospheres of Earth and other planets at varying levels of complexity. *Geophys. Model Dev.,* **11,** 843-859.

Example models that can be configured with Isca:

1. The dry model of Held and Suarez, 1994: A Proposal for the Intercomparison of the Dynamical Cores of Atmospheric General Circulation Models. *BAMS,* **75,** 1825-1830.
2. The idealised grey-radiation aquaplanet moist GCM of Frierson et al., 2006: A gray-radiation aquaplanet moist GCM. Part I: Static Stability and Eddy Scale. *J. Atmos. Sci.,* **63,** 2548-2566.

3. MiMA (Model of an idealized Moist Atmosphere), described in Jucker et al., 2017: Untangling the Annual Cycle of the Tropical Tropopause Layer with an Idealized Moist Model. *J. Climate,* **30,** 7339-7358.

A few example studies that could be replicated using Isca:

1. Kaspi and Showman, 2015: Atmospheric dynamics of terrestrial exoplanets over a wide range of orbital and atmospheric parameters. *Astrophys. J.,* **804,** 60.
2. O'Gorman and Schneider, 2008: The hydrological cycle over a wide range of climates simulated with an idealized GCM. *J. Climate,* **21,** 5797-5806.
3. Maroon et al., 2016: The Precipitation Response to an Idealized Subtropical Continent. *J. Climate,* **29,** 4543-4564.
4. Mitchell and Vallis, 2010: The transition to superrotation in terrestrial atmospheres. *J. Geophys. Res.,* **115,** E12008.

References

Frierson, D. M. W., 2007: The Dynamics of Idealized Convection Schemes and Their Effect on the Zonally Averaged Tropical Circulation. *J. Atmos. Sci.*, **64**, 1959.

Frierson, D. M. W., I. M. Held, and P. Zurita-Gotor, 2006: A Gray-Radiation Aquaplanet Moist GCM. Part I: Static Stability and Eddy Scale. *J. Atmos. Sci.*, **63**, 2548–2566.

Held, I. M., and M. J. Suarez, 1994: A Proposal for the Intercomparison of the Dynamical Cores of Atmospheric General Circulation Models. *BAMS*, **75**, 1825–1830.

Vallis, G. K., and Coauthors, 2018: Isca, v1.0: a framework for the global modelling of the atmospheres of earth and other planets at varying levels of complexity. *Geosci. Model Dev.*, **11 (3)**, 843–859, URL `https://www.geosci-model-dev.net/11/843/2018/`.