TODO for thermal hydraulics:

1. Properly handle setup\_bcs in channelarray object to update channel bcs channel-by-channel
2. Have solve\_TH in channelarray object to properly solve channel thermal hydraulics based on either equal pressure BC (‘pressure\_method’) or by ratio of total flowrate bc (‘ratio\_method’)

TODO for tracer handling:

1. Update solve\_tracer() to include channel\_conditions dict updating at the end
2. Update old\_to\_most\_recent to include tracer handling
3. Update save\_data to include tracer handling
4. Update set\_bcs() to include tracer handling
5. Update channel interface update\_interface\_conditions() to include tracer handling
6. Update ChannelArray to handle proper integration of tracer outgoing/incoming fluxes
7. Set\_bcs in channelarray for tracer handling?
8. Make def add\_tracer\_to\_channel for channelarray object

**Tracer Guide / how to / order of operations**

1. Run add\_tracer\_to\_channel() to add a tracer to this channel object.
   1. Adds kernels and BC’s
   2. BC takes on user defined value from phi.
   3. Note that BC’s need to be updated/iterated during sim. Use channel.tracer\_bcs[tracer\_name].phi
2. Solve tracer equations by running solver\_tracer()
   1. Updates channel\_conditions[“tracers\_in”] (value from bc)
   2. Updates channel\_conditions[“tracers\_out”] (value at index [-1])
3. ChannelInterface() -> update\_interface\_conditions is called to update the bc for the next channel in the loop
   1. Take channel\_conditions
      1. If outgoing leads to a Channel then use channel\_conditions[‘tracers\_out’]
      2. If outgoing leads to a ChannelArray then use get\_outlet\_conditions to compute weighted tracer outlet value based on weighted integral of tracers divided by volumetric flow rate at outlet.