ForCEPSS – CheatSheet

Matthias A.F. Gsell
matthias.gsell@medunigraz.at

August 2024

2 Experimental dictionary template

A JSON[1] dictionary of an experiment is structured in five main sections as illustrated below. Each section is further sub-structured as outlined in Sec. 2.1-2.5.

```
{
  "functions":       null OR object (see Sec. 2.1),
  "configurations":  null OR object (see Sec. 2.2),
  "solver_setup":    null OR object (see Sec. 2.3),
  "electrodes":      null OR object (see Sec. 2.4),
  "protocols":       null OR object (see Sec. 2.5),
}
```

2.1 *functions*-object

```
"functions": {
  "version":  2,
  "calibration": {
    "description":  null OR value(type=str, desc="descriptive string"),
    "tuning_dir":   value(type=str, desc="path to the tuning directory", default="./tuning")
  },
  "definitions": {
    "func_active": {  # active EP function example
      "ionic_model": {
        "model":        value(type=str, choices="run 'bench --list-imps'", desc="name of ionic model"),
        "model_par":    null OR value(type=str, desc="ionic model parameters"),
        "plugins":      null OR value(type=str, desc="name of plugins to use"),
        "plugins_par":  null OR value(type=str, desc="plugin parameters"),
        "initialization":  null OR {
          "num_cycles":       value(type=int, desc="number of initialization cycles"),
          "bcl":              value(type=real, unit=ms, desc="basic cycle lenght"),
          "init":             null OR value(type=str, desc="path to initialization file"),
          "apdres_file":      null OR value(type=str, desc="path to APD restitution file"),
          "apdres_protocol":  null OR value(type=str, desc="name of APD restitution protocol")
        }
      },
      "conductivity": {
        "gil":       value(type=real, unit=S/m, desc="intracellular longitudinal conductivity"),
        "gel":       value(type=real, unit=S/m, desc="extracellular longitudinal conductivity"),
        "git":       value(type=real, unit=S/m, desc="intracellular transverse conductivity"),
        "get":       value(type=real, unit=S/m, desc="extracellular transverse conductivity"),
        "gin":       value(type=real, unit=S/m, desc="intracellular normal conductivity"),
        "gen":       value(type=real, unit=S/m, desc="extracellular normal conductivity"),
        "surf2vol":  value(type=real, unit=1/µm, desc="surface-to-volume ratio")
      },
      "conduction_velocity": {
        "reference": {  # this is the target velocity to which the conductivity should be tuned
          "vf":  value(type=real, unit=m/s, desc="conduction velocity in fiber direction"),
          "vs":  value(type=real, unit=m/s, desc="conduction velocity in sheet direction"),
          "vn":  value(type=real, unit=m/s, desc="conduction velocity in normal direction")
        },
        "measured":  null OR {  # this is the velocity measured with current conductivity values
          "vf":  value(type=real, unit=m/s, desc="conduction velocity in fiber direction"),
          "vs":  value(type=real, unit=m/s, desc="conduction velocity in sheet direction"),
          "vn":  value(type=real, unit=m/s, desc="conduction velocity in normal direction")
        }
      }
    },
    "func_passive": {  # passive EP function example
      "conductivity":  value(type=real, unit=S/m, desc="isotropic conductivity")
    },
    ...
  }
}
```

[1] See also https://www.json.org/json-en.html

2.2 *configurations*-object

```
"configurations": {
  "version":  2,
  "definitions": {
    "config_0": {
      "tags":  list(type=int, desc="list of element tags that have the same EP function"),
      "func":  value(type=str, desc="choices are the keys of the 'functions.definitions' block")
    },
    "config_1": {
      "tags":  list(type=int, desc="list of element tags that have the same EP function"),
      "func":  value(type=str, desc="choices are the keys of the 'functions.definitions' block")
    },
    ...
  }
}
```

2.3 *solver_setup*-object

```
"solver_setup": {
  "version":       2,
  "source_model":  value(type=str, choices={monodomain,bidomain,pseudo_bidomain}, default=monodomain),
  "beqm":          value(type=bool, default=true, desc="use bidomain-equivalent-monodomain conductivity tensor if true"),
  "fixge":         value(type=bool, default=true, desc="fix extracellular conductivity, tune intracellular only if true"),
  "dt":            value(type=real, unit=µs, default=25.0, desc="time step size"),
  "dx":            value(type=real, unit=µm, default=250.0, desc="average mesh resolution"),
  "dtsubstep":     value(type=int, min=1, max=10, default=1, desc="number of ODE cycles per `dt`"),
  "lumping":       value(type=bool, default=false, desc="enable/disable mass-lumping in solver"),
  "opsplit":       value(type=bool, default=true, desc="enable/disable operator splitting in solver"),
  "tolerance":     value(type=real, default=1.0e-6, desc="error tolerance"),
  "ts":            value(type=int, choices={0,1,2}, default=1, desc="time-stepping scheme")
}
```

2.4 *electrodes*-object

```
"electrodes": {
  "version":  2,
  "definitions": {
    "vtxlist_electrode": {
      "type":     "vtxlist",
      "vtxfile":  value(type=str, desc="path to vtx file including file extension ('.vtx')")
    },
    "vtxdata_electrode": {
      "type":     "vtxdata",
      "vtxdata":  list(type=int)
    },
    "tag_electrode": {
      "type":  "tag",
      "tag":   value(type=int, desc="element tag")
    },
    "sphere_electrode": {
      "type":    "sphere",
      "p0":      list(type=int, unit=µm, length=3, desc="XYZ coordinate of center"),
      "radius":  value(type=real, unit=µm, desc="radius of the sphere")
    },
    "cylinder_electrode": {
      "type":    "cylinder",
      "p0":      list(type=int, unit=µm, length=3, desc="XYZ coordinate of start of main axis"),
      "p1":      list(type=int, unit=µm, length=3, desc="XYZ coordinate of end of main axis"),
      "radius":  value(type=real, unit=µm, desc="radius of the sphere")
    },
    "block_electrode": {
      "type":  "block",
      "p0":    list(type=int, unit=µm, length=3, desc="XYZ coordinate of first corner point"),
      "p1":    list(type=int, unit=µm, length=3, desc="XYZ coordinate of second corner point")
    },
    ...
  }
}
```

2.5 *protocols*-object[2]

```
"protocols": {
  "version":  2,
  "apd_restitution": {  # define APD restitution pacing protolocs in this section
    "apd_protocol_0": {
      "type":       value(type=str, choices={S1S2,dynamic}, desc="type of pacing protocol"),
      "bcl":        value(type=real, unit=ms, desc="basic cycle length"),
      "CI0":        value(type=real, unit=ms, desc="shortest S2 coupling interval"),
      "CI1":        value(type=real, unit=ms, desc="S1 cycle length and longest S2 coupling interval"),
      "pp_beats":   value(type=int, desc="number of beats preceding premature one"),
      "pm_beats":   value(type=int, desc="number of prepacing beats before starting protocol"),
      "pm_dec":     value(type=float, unit=ms, desc="decrement for time interval from 'CI1' to 'CI0'")
```

```
    },
    ...
  },
  "prepacing": {  # define pre-pacing protolocs in this section
    "pp_protocol_0": {
        "propagation":   value(type=str, choices={rd,ek}, desc="propagation model to use"),
        "num_cycles":    value(type=int, min=0, max=3, desc="number of prepacing cycles"),
        "bcl":           value(type=read, unit=ms, desc="basic cycle length"),
        "electrodes":    list(type=str, desc="choices are the keys of the 'electrodes.definitions' block"),
        "rel_timings":   list(type=real, unit=ms, length=length of electrodes list above, desc="pacing time for each electrode"),
        "lat_file":      null OR value(type=str, desc="path to LAT file"),
        "restart":       null OR value(type=str, desc="path to restart file")
    },
    ...
  }
}
```

All green keywords are exemplary and can be defined by the user. Red dots indicate that further entries can be defined within the corresponding block. The values of the individual entries are either predefined keywords, single values, or lists of values. Predefined keywords are for example the type of an electrode,

```
"vtxlist", "vtxdata", "tag", "sphere", "cylinder", "block"
```

and are under quotation marks. Single values are declared by the `value` keyword, including the type (`type`) and other optional specifications such as a unit (`unit`), a predefined list of choices (`choices`), a short description (`desc`), or a range of validity (`min`, `max`). A list of values is specified by the keyword `list` including the type of its entries (`type`). For lists with a defined length, the keyword `length` is specified.

[2] See also https://opencarp.org/documentation/examples/01_ep_single_cell/02b_apd_restitution