VICE: Versatile Integrator for Chemical Evolution

User's Guide

Version 1.0.0

James W. Johnson & David H. Weinberg
The Ohio State University, Department of Astronomy, 140 W. 18th Ave., Columbus, OH, 43204

Quick Links

1) Command Line

We include a command-line entry with VICE, allowing users to run mathematically simple evolutionary models directly from the linux shell. While the command-line capabilities of VICE are useful for their ease, VICE is severely limited when ran in this manner in comparison to its capabilities when opened in a python interpreter. In this environment, users are limited to their default nucleosynthetic yield settings and smooth evolutionary models. In python, VICE allows users to construct arbitrary functions of time to describe many evolutionary parameters, allowing for the simulation of chemical enrichment under much more complex parameter spaces.

2) The VICE dataframe

VICE provides a dataframe which can be indexed either by a column label (i.e. with a python variable of type str) or by row number (i.e. with a python variable of type int). For ease of use, indexing a VICE dataframe by column label is case-insensitive by construction.

Users can turn any python dictionary into a VICE dataframe by passing it directly to vice.dataframe. There are several of these structures that we include with VICE which are designed to behave in a specific manner for various reasons specific to each case. We refer users to the documentation of each individual dataframe for further details on how they vary.

All VICE dataframes have the following functions:

2.1) vice.dataframe.keys()
Returns a list of the dataframe keys in their lower-case format. This is analagous to the keys function for python dictionaries.

Example Code
```
>>> import vice
>>> example = vice.dataframe({“oNe”:  1, “TwO”: 2})
>>> example.keys()
[“two”, “one”]
```

2.2) vice.dataframe.todict() Returns a python dictionary analog of the dataframe. The keys to the dataframe will be entirely lower-case strings.

Example Code
```
>>> import vice
>>> example = vice.dataframe({“oNe”:  1, “TwO”: 2})
>>> example.todict()
{“one”:  1, “two”:  2}
```

Back to top

2.3) vice.atomic_number

A VICE dataframe containing the number of protons in the nucleus of each of VICE's recognized elements. By design, this dataframe does not support item assignment.

Example Code

```
>>> import vice
>>> vice.atomic_number["fe"]
26
>>> vice.atomic_number["ni"]
28
>>> vice.atomic_number["au"]
79
```

Back to top

2.4) vice.ccsne_yields

A VICE dataframe containing the current yield settings from core collapse supernovae. This dataframe is customizable and allows users to pass callable python functions, which will be interpreted as functions of metallicity Z. At the time an instance of the singlezone class is ran, the nucleosynthetic yield settings adopted by the simulation for each element are pulled from here.

Example Code

```
>>> import vice
>>> vice.ccsne_yields["n"]
0.000578
>>> vice.ccsne_yields["n"]
>>> vice.ccsne_yields["n"] - lambda z:  0.005 * (z / 0.014)
>>> vice.ccsne_yields["n"]
<function __main__.<lambda>>
```

2.4.1) vice.ccsne_yields.factory_defaults()

Revert the current nucleosynthetic yield settings for core collapse supernovae to their original defaults (when VICE was first installed). This will not save these settings as the new defaults; that can be achieved by calling vice.ccsne_yields.save_defaults() immediately following this function.

2.4.2) vice.ccsne_yields.restore_defaults()

Revert the current nucleosynthetic yield settings for core collapse supernovae to their current defaults (which may not be the original defaults). This will not save these settings as the new defaults; that can be achieved by calling vice.ccsne_yields.save_defaults immediately following this function.

2.4.3) vice.ccsne_yields.save_defaults()

Save the current nucleosynthetic yield settings for core collapse supernovae as defaults. Regardless of future changes in the user's current python interpreter, calling this function will make it so that the current settings are what VICE adopts upon import.

Back to top

2.5) vice.sneia_yields

2.5) vice.sneia_yields
A VICE dataframe containing the current nucleosynthetic yield settings for type Ia supernovae. This dataframe is customizable but does not allow users to pass callable functions.

Example Code

```
       >>> import vice
>>> vice.sneia_yields["fe"]
0.00258
>>> vice.sneia_yields["fe"] = 0.0017
>>> vice.sneia_yields["fe"]
0.0017
>>> vice.sneia_yields["fe"] = lambda z:  0.0017 * (z / 0.014)
TypeError:  This dataframe does not support functional attributes.
```

2.5.1) vice.sneia_yields.factory_defaults()
Revert the current nucleosynthetic yield settings for type Ia supernovae to their original defaults (when VICE was first installed). This will not save these settings as the new defaults; that can be achieved by calling vice.sneia_yields.save_defaults immediately following this function.

2.5.2) vice.sneia_yields.restore_defaults()
Revert the current nucleosynthetic yield settings for type Ia supernovae to their current defaults (which may not be the original defaults). This will not save these settings as the new defaults; that can be achieved by calling vice.sneia_yields.save_defaults immediately following this function.

2.5.3) vice.sneia_yields.save_defaults()
Save the current nucleosynthetic yield settings from type Ia supernovae as defaults. Regardless of future changes in the user's current python interpreter, calling this functions will make it so that the current settings are what VICE adopts upon import.

Back to top

2.6) `vice.solar_z`

A `VICE dataframe` containing the abundance by mass of elements in the sun. Solar abundances are derived from Asplund et al. (2009), and have been converted into a mass fraction via:

$$Z_{\mathrm{x},\odot} = \mu_{\mathrm{x}} X_{\odot} 10^{(X/H)-12} \tag{1}$$

where μ_{x} is the mean molecular weight of the element in amu, $X_{\odot}$ is the solar hydrogen abundance by mass, and $(X/H) = \log_{10}(N_{\mathrm{x}}/N_{\mathrm{H}}) + 12$, which is what Asplund et al. (2009) reports. For these calculations we adopt $X_{\odot} = 0.73$ also from Asplund et al. (2009).

This `dataframe` does not support user customization.

Example Code

```
>>> import vice
>>> vice.solar_z["o"]
0.00572
>>> vice.solar_z["fe"]
0.00129
>>> vice.solar_z["fe"] = 0.0014
TypeError:  This dataframe does not support item assignment.
```

Back to top

2.7) `vice.sources`

A `VICE dataframe` containing strings denoting what astronomers generally believe to be the dominant enrichment channels for each element. These are included purely for convenience. The fields of this `dataframe` for each element are adopted from **?**.

Example Code

```
>>> import vice
>>> vice.sources["o"]
["CCSNE"]
>>> vice.sources["fe"]
["CCSNE", "SNEIA"]
>>> vice.sources["sr"]
["CCSNE", "AGB"]
>>> vice.sources["au"]
["AGB", "NSNS"]
>>> vice.sources["au"] = ["CCSNE", "AGB"]
TypeError:  This dataframe does not support item assignment.
```

Back to top

3.1) vice.agb_yield_grid

Obtain the stellar mass-metallicity grid of fractional nucleosynthetic yields from asymptotic giant branch (AGB) stars. VICE includes yields from Karakas (2010) and Cristallo et al. (2011), allowing users the choice of which to adopt in their simulations.

Signature: `vice.agb_yield_grid(element, study = "cristallo11")`

Args

- `element` (Type: `str`)

  The element to obtain the yield grid for.

- `study` (Default: "cristallo11"; Type: `str`)

  A string (case-insensitive) denoting which AGB yield study to pull the yield table from. In its current version, only "karakas10" and "cristallo11" are the recognized values, denoting the Karakas (2010) and Cristallo et al. (2011) studies.

Returns

A 3-element python list

- `returned[0]`: A 2-D python list of yields that should be indexed via `arr[mass_index][z_index]`
- `returned[1]`: A python list containing the masses in $M_\odot$ that the yield grid is sampled on.
- `returned[2]`: A python list containing the metallicities by mass that the yield grid is sampled on.

Example Code

```
>>> import vice
>>> y, m, z = vice.agb_yield_grid("sr")
>>> m
[1.3, 1.5, 2.0, 2.5, 3.0, 4.0, 5.0, 6.0]
>>> z
[0.0001, 0.0003, 0.001, 0.002, 0.003, 0.006, 0.008, 0.01, 0.014, 0.02]
>>> # the fractional yield from 1.3 Msun stars at Z = 0.001
>>> y[0][2]
2.32254e-09
```

Back to top

3.2) vice.fractional_cc_yield

Calculate an IMF-integrated fractional nucleosynthetic yield of a given element from core-collapse supernovae. VICE has built-in functions which implement Gaussian quadrature to evaluate these integrals numerically. See VICE's science documentation for mathematical details.

Signature: vice.fractional_cc_yield(element, study = "LC18", MoverH = 0, rotation = 0, IMF = "kroupa", method = "simpson", lower = 0.08, upper = 100, tolerance = 1e-3, Nmin = 64, Nmax = 2e8)

Args

- element (Type: str)

  The element to calculate the IMF-integrated fractional yield for.

- study (Default: "LC18"; Type: str)

  A keyword (case-insensitive) denoting which study to adopt the yield from.

  Keywords and their associated studies:

  "LC18": Limongi & Chieffi (2018), ApJS, 237, 13

  "CL13": Chieffi & Limongi (2013), ApJ, 764, 21

  "CL04": Chieffi & Limongi (2004), ApJ, 608, 405

  "WW95": Woosley & Weaver (1995), ApJ, 101, 181

- MoverH (Default: 0; Type: real number)

  The total metallicity [M/H] of the exploding stars. There are only a handful of metallicities recognized by each study, and VICE will raise a ValueError if this value is not one of them.

  Keywords and their associated metallicities:

  "LC18": [M/H] = -3, -2, -1, 0

  "CL13": [M/H] = 0

  "CL04": [M/H] = -inf, -4, -2, -1, -0.37, 0.15

  "WW95": [M/H] = -inf, -4, -2, -1, 0

- rotation (Default: 0; Type: real number)

  The rotational velocity $v_{\rm rot}$ of the exploding stars in km/s. There are only a handful of rotational velocities recognized by each study, and VICE will raise a ValueError if this value is not one of them.

  Keywords and their associated rotational velocities in km/s:

  "LC18": $v_{\rm rot}$ = 0, 150, 300

  "CL13": $v_{\rm rot}$ = 0, 300

  "CL04": $v_{\rm rot}$ = 0

  "WW95": $v_{\rm rot}$ = 0

- IMF (Default: "kroupa"; Type: str)

  The stellar initial mass function (IMF) to adopt. This must be either "kroupa" for the Kroupa (2001) IMF or "salpeter" for the Salpeter (1955) IMF (case-insensitive).

- method (Default: "simpson"; Type: str)

  The method of quadrature. See section XX.XX of VICE's science documentation for details.

  Recognized methods:

  "simpson"

"trapezoid"

"midpoint"

"euler"

- `lower` (Default: 0.08; Type: real number)

  The lower mass limit on star formation in solar masses.

- `upper` (Default: 100; Type: real number)

  The upper mass limit on star formation.

- `tolerance` (Default: 10^{-3}; Type: real number)

  The numerical tolerance. The subroutines implementing Gaussian quadrature in `VICE` will not return a result until the fractional change between two successive integrations is smaller than this value.

- `Nmin` (Default: 64; Type: integer)

  The minimum number of bins in quadrature.

- `Nmax` (Default: 2×10^8; Type: integer)

  The maximum number of bins in quadrature. Included as a failsafe against non-convergent solutions.

Returns

A 2-element `python` list.

- `returned[0]`: The numerically determined solution, with an estimated precision below the specified tolerance provided the solution converges within the maximum allowed number of bins in quadrature.
- `returned[1]`: The estimated fractional error.

Example Code

```
>>> import vice
>>> y, err = vice.fractional_cc_yield("o")
>>> y
0.00564325235503O168
>>> err
4.137197161389483e-06
>>> y, err = vice.fractional_cc_yield("mg", study = "CL13")
>>> y
0.000496663271667762
```

Back to top

3.3) vice.fractional_ia_yield

Calculate an IMF-integrated fractional nucleosynthetic yield of a given element from type Ia supernovae. Unlike `vice.fractional_cc_yield`, this function does not require numerical quadrature.

Signature: `vice.fractional_ia_yield(element, study = "seitenzahl13", model = "N1", n = 2.2 × 10^-3)`

Args

- `element` (Type: `str`)

  The symbol of the element to calculate the yield for (case-insensitive).

- `study` (Default: `"seitenzahl13"`; Type: `str`)

  A keyword (case-insensitive) denoting which study to adopt the yield from.

  Keywords and their associated studies:

  `"seitenzahl13"`: Seitenzahl et al. (2013), MNRAS, 429, 1156

  `"iwamoto99"`: Iwamoto et al. (1999), ApJ, 124, 439

- `model` (Default: `"N1"`; Type: `str`)

  The model from the associated study to adopt.

  Keywords and their associated models:

  `"seitenzahl13"`: N1, N3, N5, N10, N40, N100H, N100, N100L, N150, N200, N300C

  `"iwamoto99"`: W7, W70, WDD1, WDD2, WDD3, CDD1, CDD2

- `n` (Default: 2.2×10^{-3}; Type: real number)

  The average number of type Ia supernovae produced per unit stellar mass formed N_{Ia}/M_*. This parameter has units $M_\odot^{-1}$. We recommend the default value of 2.2×10^{-3} $M_\odot^{-1}$ in accordance with Maoz & Mannucci (2012), PASA, 29, 447.

Returns

The IMF-integrated yield. Unlike `vice.fractional_cc_yield`, there is no associated numerical error with this function, because the solution is analytic.

Example Code

```
>>> import vice
>>> vice.fractional_ia_yield("fe")
0.0025825957080000002
>>> vice.fractional_ia_yield("ca")
8.935489894764334e-06
>>> vice.fractional_ia_yield("ni")
0.00016576890932800003
```

Back to top

3.4) vice.history

Read in the part of a simulation's output that records the time-evolution of the ISM metallicity.
Note: For an output under a given name, this file will be stored at *name*.vice/history.out, and it is a simple ascii text file with a comment header detailing each column. By storing the output in this manner, we allow user's to analyze the results of VICE simulations in languages other than python.

Signature: vice.history(name)

Args

- name (Type: str)

  The name of the output to read the history from as a string, with or without the '.vice' extension.

Returns
A VICE dataframe containing the time in Gyr, gas stellar masses in $M_\odot$, star formation and infall rates in $M_\odot$ yr^{-1}, inflow and outflow metallicities of each element, gas-phase mass and metallicities of each element, and every [X/Y] combination of abundance ratios for each output timestep.

Example Code

```
>>> import vice
>>> hist = vice.history("example")
>>> hist.keys()
["z(fe)",
"mass(fe)",
"[o/fe]",
"z_in(sr)",
"z_in(fe)",
"z(sr)",
"[sr/fe]",
"z_out(o)",
"mgas",
"mass(sr)",
"z_out(sr)",
"time",
"sfr",
"z_out(fe)",
"eta_0",
"[o/sr]",
"z(o)",
"[o/h]",
"ifr",
"z_in(o)",
"ofr",
"[sr/h]",
"[fe/h]",
"r_eff",
"mass(o)",
"mstar"]
>>> print("[O/Fe] at the end of the simulation:  %.2e" % (hist["[o/fe]"][-1]))
[O/Fe] at the end of the simulation:  -3.12e-01
```

Back to top

3.5) vice.mdf

Read in the normalized stellar metallicity distribution functions at the final timestep of the simulation.

Signature: vice.mdf(name)

Args

- name (Type: str)

  The name of the simulation output to read from, with or without the '.vice' extension.

Returns

A VICE dataframe containing bin edges and the value of the normalized stellar metallicity distribution in each [X/H] abundance and [X/Y] abundance ratio. These distributions are normalized such that the integral of the distribution over all bins is equal to 1, meaning that the recorded values should be interpreted as probability densities.

Example Code

```
>>> import vice
>>> m = vice.mdf("example")
>>> m.keys()
["dn/d[sr/h],",
"dn/d[sr/fe],"
"bin_edge_left,"
"dn/d[o/h],"
"dn/d[o/fe],"
"dn/d[fe/h],"
"bin_edge_right,"
"dn/d[o/sr]"]
>>> print("dn/d[O/Fe] in 65th bin:  %.2e" % (m["dn/d[o/fe]"][65]))
dn/d[O/Fe] in 65th bin:  1.41e-01
>>> print("65th bin edges:  %.2e, %.2e" % (m[65]["bin_edge_left"], m[65]["bin_edge_right"]))
65th bin edges:  2.50e-01, 3.00e-01
```

Back to top

3.6) vice.mirror

Obtain an instance of the vice.singlezone class given only an instance of the vice.output class. The returned singlezone object will have the same parameters as that which produced the output, allowing re-simulation with whatever modifications the user desires.

Signature: vice.mirror(output_obj)

Args

- output_obj (Type: vice.output)

  Any vice.output object.

Returns
A vice.singlezone object with the same attributes as that which produced the given output.

Example Code

```
>>> import vice
>>> out = vice.output("example")
>>> new = vice.mirror(out)
>>> new.settings()
Current Settings:
=================
tau_ia ---------> 1.5
recycling ------> continuous
z_solar --------> 0.014
enhancement ----> 1.0
agb_model ------> cristallo11
ria ------------> plaw
delay ----------> 0.15
imf ------------> kroupa
smoothing ------> 0.0
schmidt_index --> 0.5
eta ------------> 2.5
zin ------------> 0.0
schmidt --------> False
elements -------> (u'fe', u'sr', u'o')
MgSchmidt ------> 6000000000.0
func -----------> <function _DEFAULT_FUNC at 0x1109e06e0>
dt -------------> 0.01
tau_star -------> 2.0
name -----------> onezonemodel
m_lower --------> 0.08
m_upper --------> 100.0
Mg0 ------------> 6000000000.0
mode -----------> ifr
bins -----------> [-3, -2.95, -2.9, ...  , 0.9, 0.95, 1]
>>> # this reruns the simulation
>>> import numpy as np
>>> new.run(np.linspace(0, 10, 1001))
```

Back to top

3.7) vice.single_ia_yield

Lookup the mass yield of a given element from a single instance of a type Ia supernova as determined by a given study and explosion model.

Signature: `vice.single_ia_yield(element, study = "seitenzahl13", model = "W7")`

Args

- `element` (Type: `str`)

  The element to look up the yield for.

- `study` (Default: `"seitenzahl13"`; Type: `str`)

  A keyword (case-insensitive) denoting which study to adopt the yield from.

  Keywords and their associated studies:

  `"seitenzahl13"`: Seitenzahl et al. (2013), MNRAS, 429, 1156

  `"iwamoto99"`: Iwamoto et al. (1999), ApJ, 124, 439

- `model` (Default: `"N1"`; Type: `str`)

  The model from the associated study to adopt (case-insensitive).

  Keywords and their associated models:

  `"seitenzahl13"`: N1, N3, N5, N10, N40, N100H, N100, N100L, N150, N200, N300C

  `"iwamoto99"`: W7, W70, WDD1, WDD2, WDD3, CDD1, CDD2

Returns

The mass yield of the given element in solar masses as determined for the specified model from the specified study.

Example Code

```
>>> import vice
>>> vice.single_ia_yield("fe")
1.17390714
>>> vice.single_ia_yield("fe", study model = "W70")
0.77516
>>> vice.single_ia_yield("ni", model = "N100L")
0.0391409000000526
```

Back to top

3.8) vice.single_stellar_population

Simulate the nucleosynthesis of a given element from a single star cluster of given mass and metallicity. This does not take into account galactic evolution - whether or not it is depleted from inflows or ejected in winds is not considered. Only the mass of the given element produced by the star cluster is determined.

Signature: vice.single_stellar_population

- element (Type: str)

  The symbol of the element to simulate the enrichment for (case-insensitive)

- mstar (Default: 10^6 $M_\odot$; Type: real number)

  The total mass of the star cluster in solar masses when it is born (at time = 0).

- Z (Default: 0.014; Type: real number)

  The metallicity by mass (i.e. the mass fraction of elements heavier than helium) of the stars in the cluster.

- time (Default: 10; Type: real number)

  The amount of time in Gyr to run the simulation for.

- dt (Default: 0.01; Type: real number)

  The size of each timestep in Gyr.

- m_upper (Default: 100; Type: real number)

  The upper mass limit on star formation in solar masses.

- m_lower (Default: 0.08; Type: real number)

  The lower mass limit on star formation in solar masses.

- IMF (Default: "kroupa"; Type: str)

  The stellar initial mass function (IMF) to assume as a string (case-insensitive). Currently only "kroupa" for the Kroupa (2001), MNRAS, 322, 231 IMF and "salpeter" for the Salpeter (1955), ApJ, 121, 161 IMF are recognized.

- RIa (Default: "plaw"; Type: str or callable function)

  The delay-time distribution $R_{\rm Ia}(t)$ to adopt, where time is in Gyr. VICE will automatically normalize any function that is passed. Alternatively, VICE has built-in "plaw" (power-law, $\propto t^{-1.1}$) and "exp" (exponential, $\propto e^{-t/1.5\ \rm Gyr}$) delay-time distributions.

- delay (Default: 0.15; Type: real number)

  The minimum delay time for the onset of type Ia supernovae in Gyr.

- agb_model (Default: "cristallo11"; Type: str)

  A keyword (case-insensitive) denoting which table of nucleosynthetic yields from AGB stars to adopt.

  Recognized keywords and their associated studies:

  "cristallo11": Cristallo et al. (2011), ApJS, 197, 17

  "karakas10": Karakas (2010), MNRAS, 403, 1413

Returns

A 2-element python list.

- returned[0]: A python list containing the net mass of the given element produced by the star cluster at each timestep. Units are solar masses.

- `returned[1]`: A `python list` containing the times in Gyr corresponding to each mass yield.

Example Code

```
>>> import vice
>>> mass, times = vice.single_stellar_population("sr", mstar = 1e6, Z = 0.008)
>>> # Net strontium yield of a 1e6 Msun star cluster w/metallicity Z = 0.008
>>> print("%.2e Msun" % (mass[-1]))
4.81e-02 Msun
>>> mass, times = vice.single_stellar_population("fe", mstar = 1e6)
>>> # Net iron yield of a 1e6 Msun star cluster w/metallicity Z = 0.014
>>> print("%.2e Msun" % (mass[-1]))
2.68e+03 Msun
```

Back to top

4) Classes

`vice.integrator` (class)

Runs simulations of chemical enrichment under the single-zone approximation for user specified parameters. The organization structure of this class is simple; every attribute encodes information on a given galaxy evolution parameter. The only function in this class that the user has access to is the `run` function, which runs the simulation over the current settings.

Signature: `vice.integrator(self, name = "onezonemodel", func = _globals._DEFAULT_FUNC, mode = "ifr", elements = ["fe", "sr", "o"], imf = "kroupa", eta = 2.5, enhancement = 1, Zin = 0, recycling = "continuous", bins = _globals._DEFAULT_FUNC, delay = 0.15, dtd = "plaw", Mg0 = 6.0e9, smoothing = 0.0, tau_ia = 1.5, tau_star = 2.0, dt = 0.01, schmidt = False, schmidt_index = 0.5, MgSchmidt = 6.0e9, m_upper = 100, m_lower = 0.08, Z_solar = 0.014, agb_model = "cristallo11")`

Attributes

- `name` (Default: `"onezonemodel"`; Type: `str`)

  The name of the integrator. The output will be stored in a directory under this name with the extension `".vice"`. This can also be of the form `"/path/to/directory/name"` and the output will be stored there. The user need not interact with any of the output files; the `vice.output` object is designed to read in all of the results automatically.

  Outputs are handled in this format because it allows users to open the files in languages other than `python`, as the simulation results are recorded in pure `ascii` text. Forcing a `".vice"` extension on the name of the directory allows users to run commands in a linux kernel over all outputs in a directory via `<command> *.vice`. Within each `VICE` output are two `".out"` files; `"history.out"` contains the time-evolution of the ISM metallicity and `"mdf.out"` contains the normed stellar metallicity distribution function. The three `".config"` output files contain `python` dictionaries stored in a `pickle` with the information to reconstruct the yield settings and `integrator` settings from the output.

- `func` (Default: `_globals._DEFAULT_FUNC`; Type: `<function>`)

  A callable `python` function of time which returns a numerical value. It must only take one parameter, which `VICE` will interpret as time in Gyr. The value returned by this function will represent either the gas infall history $\dot{M}_{\rm in}$ in $\mathrm{M}_\odot$ yr^{-1}, the star formation history in $\dot{M}_*$ in $\mathrm{M}_\odot$ yr^{-1}, or the gas supply in $\mathrm{M}_\odot$ at that time.

  The default function returns the value 9.1 at all times. `VICE` defaults to infall mode, meaning that would represent an constant infall rate of 9.1 $\mathrm{M}_\odot$ yr^{-1} if neither of these parameters were changed.

  See user's notes on functional attributes and numerical delta functions in `VICE`.

- `mode` (Default: `"ifr"`; Type: `str` - either `"ifr"`, `"sfr"`, or `"gas"` [case-insensitive])

  The interpretation of the attribute `func`. If `"ifr"` it represents the infall rate in $\mathrm{M}_\odot$ yr^{-1}; if `"sfr"` it represents the star formation history in $\mathrm{M}_\odot$ yr^{-1}; if `"gas"` it represents the gas supply in $\mathrm{M}_\odot$. The argument to `func` will always be interpreted as time in Gyr.

- `elements` (Default: ["fe", "sr", "o"]; Type: array-like - elements of type `str` [case-insensitive])

  The symbols for the elements to track the enrichment for. The more elements that are tracked, the more precisely calibrated is the total ISM metallicity at each timestep for responding to metallicity dependent nucleosynthetic yields, but the longer each simulation will take. In its current version, `VICE` simulates enrichment via core collapse supernovae, type Ia supernovae, and asymptotic giant branch stars for all 76 elements between carbon ("c") and bismuth ("bi").

  `VICE` is compatible with the `NumPy array` and `Pandas DataFrame`, but is not dependent on either package.

- `imf` (Default: "kroupa"; Type: str - either "kroupa" or "salpeter")

  The assumed stellar initial mass function (IMF). VICE currently recognizes the Kroupa (2001) and Salpeter (1955) IMFs. A future update will likely include a wider sample of IMFs.

- `eta` (Default: 2.5; Type: real number or <function>)

  The mass loading factor $\eta = \dot{M}_{\rm out}/\dot{M}_*$ (the ratio of the outflow to the star formation rate). This can also be a callable python function taking exactly one parameter, which will be interpreted as time in Gyr.

  See user's notes on functional attributes and numerical delta functions in VICE.

  If the user changes the smoothing timescale via the attribute smoothing, the relationship between the outflow rate and the star formation rate becomes more complicated. See VICE's science documentation at https://github.com/giganano/VICE/tree/master/docs or this parameter's docstring for mathematical and numerical details.

- `enhancement` (Default: 1; Type: real number or <function>)

  The ratio of the outflow to ISM metallicities. This can also be a callable python function taking exactly one parameter, which will be interpreted as time in Gyr.

  See user's notes on functional attributes and numerical delta functions in VICE.

- `Zin` (Default: 0; Type: real number, <function>, or python dictionary [case-insensitive])

  The metallicity of gas inflow. This can either be a number, which will apply to all elements, a <function> of time in Gyr which will also apply to all elements, or a python dictionary mapping elemental symbols [VICE will respond case-insensitively] to either real numbers or callable functions of time in Gyr. This allows the user to construct arbitrary functions of time for each element, allowing them to simulate the effects of infall metallicities in full generality.

  See user's notes on functional attributes and numerical delta functions in VICE.

- `recycling` (Default: "continuous"; Type: real number or str - if str, it must be "continuous" [case-insensitive])

  The cumulative return fraction r. This is the mass fraction of a single stellar population returned to the ISM as gas at the birth metallicity of the stars. By default, VICE implements continuous recycling off of a treatment of the IMF weighted by the mass of remnants as modeled by Kalirai et al. (2008). If the user specifies a numerical value, it must be between 0 and 1. In this case it will represent the instantaneous recycling parameter $r_{\rm inst}$ as in the analytic model of Weinberg et al. (2017).

  See VICE's science documentation at https://github.com/giganano/VICE/tree/master/docs or this parameter's docstring for mathematical and numerical deatils.

- `bins` (Default: _globals._DEFAULT_BINS; Type: array-like - elements are real numbers)

  The bins in each [X/H] logarithmic abundance measurements and each [X/Y] abundance ratio to sort the normed stellar metallicity distribution function into. By default, VICE sorts everything into 0.01-dex width bins between [X/H] and [X/Y] = -3 and +1.

  VICE is compatible with the NumPy array and Pandas DataFrame, but is not dependent on either package.

- `delay` (Default: 0.15; Type: real number)

  The minimum delay time in Gyr for the onset of type Ia supernovae associated with each episode of star formation.

- `dtd` (Default: "plaw"; Type: str [case-insensitive] or <function>)

  The delay-time distribution for type Ia supernovae to adopt. If type str, VICE will use built-in delay-time distributions (DTDs). These are "exp" for DTDs which decay exponentially with e-folding timescale set by the attribute tau_ia and "plaw" for a power law with index 1.1 (i.e. $R_{\rm Ia} \propto t^{-1.1}$).

  Alternatively the user may pass their own function of time as $R_{\rm Ia}(t)$. The user need not worry about normalizing their custom DTD; VICE will take care of that automatically.

  See user's notes on functional attributes and numerical delta functions in VICE.

- `Mg0` (Default: 6×10^9; Type: real number)

  The mass of the ISM gas at time $t = 0$ in $M_\odot$. This parameter only matters when the `integrator` is in infall mode. In gas mode, `func`(0) specifies the initial gas supply, and in star formation mode, it is `func`(0) $*$ `tau_star`(0).

- `smoothing` (Default: 0; Type: real number)

  The smoothing time in Gyr to adopt. This is the timescale on which the star formation rate is time-averaged before determining the outflow rate via the mass loading parameter η (attribute `eta`).

  See `VICE`'s science documentation at https://github.com/giganano/VICE/tree/master/docs or this parameter's docstring for mathematical and numerical deatils.

- `tau_ia` (Default: 1.5; Type: real number)

  The e-folding timescale in Gyr of an exponentially decaying delay-time distribution for type Ia supernovae. This parameter only matters when the attribute `dtd = "exp"`.

- `tau_star` (Default: 2.0; Type: real number or <`function`>)

  The star formation efficiency (SFE) timescale denoting the gas supply per unit star formation; $\tau_* = M_g / \dot{M}_*$ in Gyr. In obserational journal articles, this is often referred to as the "depletion time". This parameter is how the gas supply and star formation rate are determined off of one another at each timestep.

  See user's notes on functional attributes and numerical delta functions in `VICE`.

- `dt` (Default: 0.01; Type: real number)

  The timestep size in Gyr. For fine timesteps with a given ending time in the simulation, this affects the total integration time with a Δt^{-2} dependence.

- `schmidt` (Default: `False`; Type: boolean)

  A boolean describing whether or not to use an implementation of gas-dependent star formation efficiency (i.e. the Kennicutt-Schmidt Law; Schmidt, 1959; Leroy et al., 2008). At each timestep, the user-specified `tau_star`, normalization `MgSchmidt`, and `schmidt_index` determine the star formation efficiency at that timestep via:

$$\tau_*^{-1} = \texttt{tau\_star}(t)^{-1} \left(\frac{M_g}{\texttt{MgSchmidt}} \right)^{\texttt{schmidt\_index}} \tag{2}$$

- `schmidt_index` (Default: 0.5; Type: real number)

  The power law index on star formation efficiency driven by the Kennicutt-Schmidt Law (Schmidt, 1959; Leroy et al., 2008). See `schmidt` documentation for further details, or `VICE`'s science documentation at https://github.com/giganano/VICE/tree/master/docs for mathematical and numerical details.

- `MgSchmidt` (Default: 6×10^9; Type: real number)

  The normalization of the gas supply when the star formation efficiency is driven by the Kennicutt-Schmidt Law (Schmidt, 1959; Leroy et al., 2008). In pracitce, this quantity should be comparable to a typical gas supply of the simulated galaxy so that the actual star formation efficiency at a given timestep is near the user-specified value. See documentation for the attribute `schmidt` for further details.

- `m_upper` (Default: 100; Type: real number)

  The upper mass limit on star formation in $M_\odot$.

- `m_lower` (Default 0.08; Type: real number)

  The lower mass limit on star formation in $M_\odot$.

- `Z_solar` (Default: 0.014; Type: real number)

  The metallicity of the sun by mass. We recommend the default value of 0.014 based on the findings of Asplund et al. (2009).

- agb_model (Default: "cristallo11"; Type: str - either "cristallo11" or "karakas10")

  A string denoting the keyword for the stellar mass-metallicity grid of yields from asymptotic giant branch (AGB) stars to adopt. In its current version, these are the only yield options for AGB stars recognized by VICE. Future updates will include a wider sample of built-in yield tables, construct their own yield tables, and potentially specify functions of mass and metallicity for each element.

Functions

vice.integrator.run(output_times, capture = False, overwrite = False)

Runs VICE's built-in timestep integration routines over the parameters built into the attributes of this class. Whether or not the user sets capture = True, the output files will be produced and can be read into a vice.output object at any time.

- output_times (Type: array-like, elements are real number)

  An array of times in Gyr at which the simulation should write output. This need not be sorted in any order.

- capture (Default: False; Type: Boolean)

  If True, will return a vice.output object for the results of the simulation.

- overwrite (Default: False; Type: Boolean)

  If True, will force overwrite any files under the same name as the output. If not, this acts as a halting function because VICE will stop to confirm that the output files are to be overwritten.

vice.output (class)

Reads in the output from the vice.integrator class and allows the user to access it easily in a case-insensitive manner. The results are read in automatically upon instantiating an output object.

Signature: vice.output(name)

Attributes

- name (Type: str)

  The name of the ".vice" directory containing the output of an "integrator" object. The ".vice" extension need not be specified with the name. Upon instantiating an output object, the results will be read automatically.

- ccsne_yields (Type: VICE dataframe)

  A case-insensitive dataframe containing the yield settings from core-collapse supernovae at the time the simulation was ran.

- elements (Type: tuple)

  A python tuple containing the symbols of the elements whose enrichment was tracked by the simulation.

- history (Type: VICE dataframe)

  A case-insensitive dataframe containing the time-evolution of the galaxy and its abundances, such as the star formation rate, gas mass, and abundances. This can be keyed on by either column name or line number. For example:

  ```
  >>> import vice
  >>> out = vice.output("onezonemodel")
  >>> sfr = out.history["sfr"]
  >>> ofe = out.history["[o/fE]"]
  >>> hundredth_line = out.history[100]
  ```

  where we have made deliberate case errors in the fourth line to demonstrate that the VICE dataframe is case insensitive. In the case of the final line, hundredth_line will also be a VICE dataframe.

- `mdf` (Type: `VICE dataframe`)

  A case-insensitive dataframe containing the normalized stellar metallicity distribution function at the final timestep of the simulation. This quantity represents the probability density that a random star would have a given [X/H] abundance or [X/Y] abundance ratio in the given bin. This `dataframe` also contains the bin edges that the user built-into the `integrator` to run the simulation.

  NOTE: If any [X/H] abundances or [X/Y] abundance ratios determined by `VICE` never pass within the user's specified binspace, then the associated MDF will be `NaN` at all values.

- `sneia_yields` (Type: `VICE dataframe`)

  A case-insensitive dataframe containing the yield settings from type Ia supernovae at the time the simulation was ran.

Functions

`vice.output.show(key)`

Show a plot of the given quantity referenced by the argument `key` [case-insensitive]. If this is a quantity stored in the `vice.output.history dataframe`, it will be plotted against time by default. Conversely, if it is stored in the `vice.output.mdf dataframe`, it will show the corresponding stellar metallicity distribution function.

Users can also specify an argument of the format `key1-key2` where `key1` and `key2` are elements of the `vice.output.history dataframe`. It will then plot `key1` against `key2` and show it to the user. For example:

```
>>> import vice
>>> out = vice.output("example")
# Will show the star formation rate against time in Gyr
>>> out.show("sfr")
# Will show the stellar MDF in [O/Fe]
>>> out.show("dn/d[o/fe]")
# Will show the track in the [O/Fe]-[Fe/H] plane
>>> out.show("[O/Fe]-[Fe/H]")
```

NOTE: This function is not intended to generate publication quality plots for users. It is included purely as a convenience function for users to be able to read in and immediately inspect the results of their simulations in a plot with only a few lines of code.

This is the only function included with `VICE` that is dependent on any derivative of `anaconda`, requiring `matplotlib` version ≥ 2.

User's Note on Functional Attributes in `VICE`

Functional attributes in `VICE` must be native `python` functions. Any byte-compiled function, such as `NumPy` mathemetical functions or any `python` code that has been ran through a `Cython` compiler, will produce a `TypeError`. If the user wishes to employ one of these functions, this can be achieved by simply wrapping them in a `python` function. For example, the following will produce a `TypeError`:

```
>>> import numpy as np
>>> import vice
>>> intgtr = vice.integrator()
>>> intgtr.func = np.exp
```

but the following will not:

```
>>> import numpy as np
>>> import vice
>>> intgtr = vice.integrator()
>>> def f(t):
>>>     return np.exp(t)
>>> intgtr.func = f
```

The same can be achieved with a `python lambda`.

User's Note on Numerical Delta Functions

VICE is a timestep-style integrator, and therefore, numerical delta functions can be achieved by letting a quantity take on some very high value for one timestep. If the user wishes to build a delta function into their model, they need to make sure that:

1. They let their delta function have an intrinsic finite width of at least one timestep. Otherwise, it is not guaranteed that the numerical integrator will find the delta function.
2. They have set their output times such that the integrator will write to the output file at the time of the delta function. If this is not ensured, the output will still show the behavior induced by the delta function, but not in the parameter which was meant to exhibit one.

When running an integrator, we recommend that users set it to write output at every timestep for a brief period following a numerical delta function in any parameter. This simply ensures that the output will reflect more detail following its onset. See `vice.integrator.run` docstring for details

Bibliography

Asplund M., Grevesse N., Sauval A. J., Scott P., 2009, ARA&A, 47, 481

Cristallo S., et al., 2011, ApJS, 197, 17

Kalirai J. S., Hansen B. M. S., Kelson D. D., Reitzel D. B., Rich R. M., Richer H. B., 2008, ApJ, 676, 594

Karakas A. I., 2010, MNRAS, 403, 1413

Kroupa P., 2001, MNRAS, 322, 231

Leroy A. K., Walter F., Brinks E., Bigiel F., de Blok W. J. G., Madore B., Thornley M. D., 2008, AJ, 136, 2782

Salpeter E. E., 1955, ApJ, 121, 161

Schmidt M., 1959, ApJ, 129, 243

Weinberg D. H., Andrews B. H., Freudenburg J., 2017, ApJ, 837, 183