

# VICE: Versatile Integrator for Chemical Evolution

## User's Guide

### Version 1.0.0

James W. Johnson

*The Ohio State University, Department of Astronomy, 140 W. 18th Ave., Columbus, OH, 43204*

VICE is open-source software released under the MIT license. We invite researchers and developers to use, modify, and redistribute however they see fit under the terms of the associated license. VICE's source code and installation instructions can be found at <http://github.com/giganano/VICE.git>. Usage of VICE leading to a publication should cite Johnson & Weinberg (2019, in prep).

*This documentation is intended to serve neither as a review of galactic chemical evolution modeling nor as a means of finding citations on previous work in this field. This provides nothing more and nothing less than standard documentation with instructions on how to use each component of the software.*

### Contents

#### Command Line

**dataframe**: Case-insensitive data storage

**keys**: Access dataframe keys

**todict**: Convert dataframe to dictionary

**atomic\_number**: Atomic number lookup for convenience

**solar\_z**: Solar metallicity by mass lookup for convenience

**sources**: Dominant sources of enrichment lookup for convenience

**yields**: Declaring and handling nucleosynthetic yields from various channels

**agb**: Nucleosynthetic yields from asymptotic giant branch stars

**grid**: Read the stellar mass-metallicity grid of fractional yields

**ccsne**: Nucleosynthetic yields from core collapse supernovae

**fractional**: Calculate an IMF-integrated fractional yield

**settings**: Declare nucleosynthetic yields to adopt in simulations

**CL04**: Chieffi & Limongi (2004) Nucleosynthetic Yields

**CL13**: Chieffi & Limongi (2013) Nucleosynthetic Yields

**LC18**: Limongi & Chieffi (2018) Nucleosynthetic Yields

**WW95**: Woosley & Weaver (1995) Nucleosynthetic Yields

**sneia**: Nucleosynthetic yields from type Ia supernovae

**fractional**: Calculate an IMF-integrated fractional yield

**settings**: Declare nucleosynthetic yields to adopt in simulations

**single**: Look up the yield of a given element from a single type Ia supernova

**iwamoto99**: Iwamoto et al. (1999) Nucleosynthetic Yields

**seitenzahl13**: Seitenzahl et al. (2013) Nucleosynthetic Yields

**history**: Read in the time-evolution of the ISM from the output

**cumulative\_return\_fraction**: Calculate the cumulative return fraction

**main\_sequence\_mass\_fraction**: Calculate the main sequence mass fraction

**mdf**: Read in the resulting stellar metallicity distribution function from an output

**mirror**: Obtain a **singlezone** object with the same parameters as that which produced a given output

**output**: Read in all output from an instance of the **singlezone** class

**show**: Produce a plot of a given quantity stored in the output

**single\_stellar\_population**: Simulate mass enrichment of a given element from only one population of stars

**singlezone**: Construct simulations given user-specified parameters

**run**: Run the simulation over the current parameters and yield settings

#### Exception Handling

#### User's Notes

---

## Command Line

I include a command-line entry with VICE, allowing users to run mathematically simple evolutionary models directly from the command line. For instructions on how to make use of this feature, simply run `vice --help` in a linux terminal after installing VICE. 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.

---

## **vice.dataframe**

A class meant for storing data indexed by strings in a case-insensitive manner. VICE includes several instances of this class at the global level, some of which have features specific to their instance. Users may call these dataframes as a function using parentheses rather than square brackets and it will return the same thing. If a dataframe stores array-like attributes, it may also be indexed via an integer to pull that element from each field.

### **Included Dataframes**

- [atomic\\_number](#)
- [solar\\_z](#)
- [sources](#)
- [yields.ccsne.settings](#)
- [yields.snea.settings](#)

---

## **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**

```
>>> example = vice.dataframe({"oNe": 1, "Tw0": 2})
>>> example.keys()
["two", "one"]
```

---

## **vice.dataframe.todict**

Returns a python dictionary analog of the dataframe. The keys to the dataframe will be entirely lower-case strings.

### **Example**

```
>>> example = vice.dataframe({"oNe": 1, "Tw0": 2})
>>> example.todict()
{"one": 1, "two": 2}
```

---

## **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**

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

---

## 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), ARA&A, 47, 481, and have been converted into a mass fraction via:

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

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

This dataframe does not support user customization.

### **Example**

```
>>> 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.
```

---

## **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 Johnson (2019), Science, 363, 474.

### **Example**

```
>>> 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.
```



---

## vice.yields

**Nucleosynthetic Yield Tools:** Each module stores built-in yield tables and user-presets for different enrichment channels.

### Included Modules

- [vice.yields.agb](#)
- [vice.yields.ccsne](#)
- [vice.yields.sneia](#)

---

## **vice.yields.agb**

**Asymptotic Giant Branch Star Nucleosynthetic Yield Tools:** In the current version of VICE, users are allowed to select between two tables of nucleosynthetic yields from asymptotic giant branch stars - those published by the Karakas (2010) and the Cristallo et al. (2011) studies.

### **Included Features**

- [vice.yields.agb.grid](#)

### **References**

Cristallo (2011), ApJS, 197, 17  
Karakas (2010), MNRAS, 403, 1413

---

## vice.yields.agb.grid

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

**Signature:** `vice.yields.agb.grid(element, study = "cristallo11")`

### Parameters

- `element` (Type: `str` [case-insensitive])  
The symbol of the element to obtain the yield grid for.
- `study` (Type: `str` [case-insensitive]; Default: `"cristallo11"`)  
A keyword denoting which AGB yield study to pull the yield table from.

### Keywords and their Associated Studies

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

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

### Returns

- `grid` (Type: `tuple` (2D))  
The yield grid itself; elements are tuples of fractional nucleosynthetic yields at constant stellar mass, but varying metallicity. It should be indexed with the rule: `arr[mass_index][z_index]`
- `masses` (Type: `tuple`)  
The masses in terms of the sun that the yield grid is sampled on.
- `z` (Type: `tuple`)  
The metallicities by mass Z on that the yield grid is sampled on.

### Raises

- `ValueError`
  - The study or element are not built into VICE
- `LookupError`
  - `study == "karakas10"` and the atomic number of the element is greater than or equal to 29. The Karakas (2010), MNRAS, 403, 1413 study did not report yields from AGB stars for elements heavier than nickel.
- `IOError` [Occurs only if VICE's file structure has been tampered with]
  - The parameters passed to this function are allowed but the data file is not found.

### Example

```
>>> y, m, z = vice.yields.agb.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
```

### **References**

Cristallo et al. (2011), ApJS, 197, 17  
Karakas (2010), MNRAS, 403, 1413

---

## vice.yields.ccsne

**Core Collapse Supernovae Nucleosynthetic Yield Tools:** Here users can both calculate IMF-integrated nucleosynthetic yields from core collapse supernovae as well as modify their yield settings, which VICE will adopt in their simulations. VICE has built-in tables allowing users to calculate IMF-integrated yields from the results of supernovae simulations ran by four different studies, which can also be directly imported as the yield settings.

### Included Features

- `fractional` (Type: <function>)  
Calculates IMF-integrated yields of a given element.
- `settings` (Type: dataframe)  
Stores the user's current settings for these yields.

### Built-in Yield Tables Available for Import

- [CL04: Chieffi & Limongi \(2004\)](#)
- [CL13: Chieffi & Limongi \(2013\)](#)
- [LC18: Limongi & Chieffi \(2018\)](#)
- [WW95: Woosley & Weaver \(1995\)](#)

### References

Chieffi & Limongi (2004), ApJ, 608, 405  
Chieffi & Limongi (2013), ApJ, 764, 21  
Limongi & Chieffi (2018), ApJS, 237, 13  
Woosley & Weaver (1995), ApJ, 101, 181

---

## vice.yields.ccsne.fractional

Calculates 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 section 5.2 of VICE’s science documentation at <https://github.com/giganano/VICE/tree/master/docs> for further details.

**Signature:** `vice.yields.ccsne.fractional(element, study = “LC18”, MoverH = 0, rotation = 0, IMF = “kroupa”, method = “simpson”, m_lower = 0.08, m_upper = 100, tolerance = 1e-3, Nmin = 64, Nmax = 2e8)`

### Parameters

- **element** (Type: str [case-insensitive])  
The symbol of the element to calculate the IMF-integrated fractional yield for.
- **study** (Type: str [case-insensitive]; Default: “LC18”)  
A keyword 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** (Type: real number; Default: 0)  
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 `LookupError` 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** (Type: real number; Default: 0)  
The rotational velocity  $v_{\text{rot}}$  of the exploding stars in km/s. There are only a handful of rotational velocities recognized by each study, and VICE will raise a `LookupError` if this value is not one of them.

#### Keywords and their Associated Rotational Velocities in km/s

- “LC18”:  $v_{\text{rot}}$  = 0, 150, 300
  - “CL13”:  $v_{\text{rot}}$  = 0, 300
  - “CL04”:  $v_{\text{rot}}$  = 0
  - “WW95”:  $v_{\text{rot}}$  = 0
- **IMF** (Type: str [case-insensitive]; Default: “kroupa”)  
The stellar initial mass function (IMF) to adopt. This must be either “kroupa” for the Kroupa (2001), MNRAS, 322, 231 study or “salpeter” for the Salpeter (1955), ApJ, 121, 161 study.

- 
- **method** (Type: str [case-insensitive]; Default: “simpson”)

The method of quadrature. The numerical rules implemented here are of the forms outlined in Chapter 4 of Numerical Recipes (Press, Teukolsky, Vetterling & Flannery).

#### **Recognized Methods**

- “simpson”
- “trapezoid”
- “midpoint”
- “euler”

- **m\_lower** (Type: real number; Default: 0.08)

The lower mass limit on star formation in solar masses.

- **m\_upper** (Type: real number; Default: 100)

The upper mass limit on star formation in solar masses.

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

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** (Type: integer; Default: 64)

The minimum number of bins in quadrature.

- **Nmax** (Type: integer; Default:  $2 \times 10^8$ )

The maximum number of bins in quadrature. Included as a failsafe against solutions that don’t converge numerically.

#### **Returns**

- **yield** (Type: real number)

The numerically determined solution.

- **error** (Type: real number)

The estimated fractional numerical error.

#### **Raises**

- **ValueError**

- The element is not built into VICE
- The study is not built into VICE
- The tolerance is not between 0 and 1
- $m\_lower > m\_upper$
- The IMF is not built into VICE
- The method of quadrature is not built into VICE
- $Nmin > Nmax$

- **LookupError**

- The study did not report yields at the specified metallicity.
- The study did not report yields at the specified rotational velocity.

---

- ScienceWarning

- `m_upper` is larger than the largest mass on the grid reported by the specified study. VICE extrapolates to high masses in this case.
- `study` is either “CL04” or “CL13” and the atomic number of the element is between 24 and 28 (inclusive). VICE warns against adopting these yields for iron peak elements.
- Numerical quadrature did not converge within the maximum number of allowed quadrature bins to within the specified tolerance.

### Notes

This function evaluates the solution to the following equation under the assumption that all stars above  $8M_{\odot}$  and below the upper mass limit on star formation explode as a core collapse supernova.

$$y_x^{\text{CC}} = \frac{\int_8^u M_x \frac{dN}{dM} dM}{\int_l^u M \frac{dN}{dM} dM} \quad (2)$$

where  $M_x$  is the mass of the element  $x$  produced in the supernova of a star of initial mass  $M$ , and  $dN/dm$  is the stellar IMF.

### Example

```
>>> y, err = vice.yields.ccsne.fractional("o")
>>> y
0.005643252355030168
>>> err
4.137197161389483e-06
>>> y, err = vice.yields.ccsne.fractional("mg", study = "CL13")
>>> y
0.000496663271667762
```



---

## **vice.yields.ccsne.settings**

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**

```
>>> vice.yields.ccsne.settings["n"]
0.000578
>>> vice.yields.ccsne.settings["n"] = lambda z: 0.005 * (z / 0.014)
>>> vice.yields.ccsne.settings["n"]
<function __main__.<lambda>>>
```

### **vice.yields.ccsne.settings.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.yields.ccsne.settings.save_defaults()` immediately following this function.

### **vice.yields.ccsne.settings.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.yields.ccsne.settings.save_defaults()` immediately following this function.

### **vice.yields.ccsne.settings.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.

### **Notes**

Saving functional yield parameters requires the python package `dill` (<https://pypi.org/project/dill/>). VICE will run properly without this package, but users will not be able to save default yields as functions of metallicity if they do not have `dill` installed.

---

## **vice.yields.ccsne.CL04**

**Chieffi & Limongi (2004), ApJ, 608, 405 Nucleosynthetic Yield Tools:** Importing this module will automatically set all yield settings from core collapse supernovae to the IMF-integrated yields as determined from the simulations ran by Chieffi & Limongi (2004) at  $Z = 0.02$  ( $[M/H] = 0.15$  if the solar abundance is  $Z = 0.014$ ; Asplund et al. 2009). In doing so, it will default to an upper mass limit on star formation of  $35 M_{\odot}$ . This is done to minimize numerical artifacts; this is the highest mass on the Chieffi & Limongi (2004) grid.

VICE achieves this by calling `yields.ccsne.fractional` for every element built into the software and storing the returned value in `yields.ccsne.settings`.

[CL04.set\\_params](#): Update the parameters with which the yields are calculated.

### **Notes**

By importing this module, the user does not sacrifice the flexibility of VICE's user specified yields. The fields of `vice.yields.ccsne.settings` can still be modified in whatever manner the user sees fit.

This module is not imported with the simple `import vice` statement.

### **Example**

```
>>> from vice.yields.ccsne import CL04
>>> CL04.set_params(m_lower = 0.3, m_upper = 40, IMF = "salpeter")
```

### **References**

Asplund et al. (2009), ARA&A, 47, 481

---

## **vice.yields.ccsne.CL04.set\_params**

Update the parameters with which the yields are calculated from the Chieffi & Limongi (2004) data.

### **Parameters**

- `kwargs` (varying types)  
Keyword arguments to pass to `yields.ccsne.fractional`

### **Raises**

- `TypeError`
  - The user has specified a keyword argument “study”.
- Other exceptions are raised by `yields.ccsne.fractional`

### **See Also**

[yields.ccsne.fractional](#)

### **Example**

```
>>> from vice.yields.ccsne import CL04
>>> CL04.set_params(m_lower = 0.3, m_upper = 40, IMF = “salpeter”)
```

[Back to CL04](#)

### **References**

Chieffi & Limongi (2004), ApJ, 608, 405

---

## **vice.yields.ccsne.CL13**

**Chieffi & Limongi (2013), ApJ, 764, 21 Nucleosynthetic Yield Tools:** Importing this module will automatically set all yield settings from core collapse supernovae to the IMF-integrated yields as determined from the simulations ran by Chieffi & Limongi (2013) at solar metallicity.

VICE achieves this by calling `yields.ccsne.fractional` for every element built into the software and storing the returned value in `yields.ccsne.settings`.

[CL13.set\\_params](#): Update the parameters with which the yields are calculated.

### **Notes**

By importing this module, the user does not sacrifice the flexibility of VICE's user specified yields. After importing this module, the fields of `vice.yields.ccsne.settings` can still be modified in whatever manner the user sees fit.

This module is not imported with the simple `import vice` statement.

### **Example**

```
>>> from vice.yields.ccsne import CL13
>>> CL13.set_params(m_lower = 0.3, m_upper = 40, IMF = "salpeter")
```

---

## **vice.yields.ccsne.CL13.set\_params**

Update the parameters with which the yields are calculated from the Chieffi & Limongi (2013) data.

### **Parameters**

- `kwargs` (varying types)  
Keyword arguments to pass to `yields.ccsne.fractional`

### **Raises**

- `TypeError`
  - The user has specified a keyword argument “study”.
- Other exceptions are raised by `yields.ccsne.fractional`

### **See Also**

[yields.ccsne.fractional](#)

### **Example**

```
>>> from vice.yields.ccsne import CL13
>>> CL13.set_params(m_lower = 0.3, m_upper = 40, IMF = “salpeter”)
```

[Back to CL13](#)

### **References**

Chieffi & Limongi (2013), ApJ, 764, 21

---

## **vice.yields.ccsne.LC18**

**Limongi & Chieffi (2018), ApJS, 237, 13 Nucleosynthetic Yield Tools:** Importing this module will automatically set all yield settings from core collapse supernovae to the IMF-integrated yields as determined by Limongi & Chieffi (2018) at solar metallicity.

VICE achieves this by calling `yields.ccsne.fractional` for every element built into the software and storing the returned value in `yields.ccsne.settings`.

[LC18.set\\_params](#): Update the parameters with which the yields are calculated.

### **Notes**

By importing this module, the user does not sacrifice the flexibility of VICE's user-specified yields. After importing this module, the fields of `vice.yields.ccsne.settings` can still be modified in whatever manner the user sees fit.

This module is not imported with the simple `import vice` statement.

### **Example**

```
>>> from vice.yields.ccsne import LC18
>>> LC18.set_params(m_lower = 0.3, m_upper = 40, IMF = "salpeter")
```

---

## **vice.yields.ccsne.LC18.set\_params**

Update the parameters with which the yields are calculated from the Limongi & Chieffi (2018) data.

### **Parameters**

- `kwargs` (varying types)  
Keyword arguments to pass to `yields.ccsne.fractional`

### **Raises**

- `TypeError`
  - The user has specified a keyword argument “study”.
- Other exceptions are raised by `yields.ccsne.fractional`.

### **See Also**

[yields.ccsne.fractional](#)

### **Example**

```
>>> from vice.yields.ccsne import LC18
>>> LC18.set_params(m_lower = 0.3, m_upper = 40, IMF = “salpeter”)
```

[Back to LC18](#)

### **References**

Limongi & Chieffi (2018), ApJS, 237, 17

---

## **vice.yields.ccsne.WW95**

**Woosley & Weaver (1995), ApJ, 101, 181 Nucleosynthetic Yield Tools:** Importing this module will automatically set all yield settings from core collapse supernovae to the IMF-integrated yields as determined from the simulations ran by Woosley & Weaver (1995) at solar metallicity. In doing so, it will default to an upper mass limit on star formation of  $40 M_{\odot}$ . This is done to minimize numerical artifacts; this is the highest mass on the Woosley & Weaver (1995) grid.

VICE achieves this by calling `yields.ccsne.fractional` for every element built into the software and storing the returned value in `yields.ccsne.settings`.

[WW95.set\\_params](#): Update the parameters with which the yields are calculated.

### **Notes**

By importing this module, the user does not sacrifice the flexibility of VICE's user-specified yields. After importing this module, the fields of `vice.yields.ccsne.settings` can still be modified in whatever manner the user sees fit.

This module is not imported with the simple `import vice` statement.

### **Example**

```
>>> from vice.yields.ccsne import WW95
>>> WW95.set_params(m_lower = 0.3, m_upper = 40, IMF = "salpeter")
```



---

## **vice.yields.ccsne.WW95.set\_params**

Update the parameters with which the yields are calculated from the Woosley & Weaver (1995) data.

### **Parameters**

- `kwargs` (varying types)  
Keyword arguments to pass to `yields.ccsne.fractional`

### **Raises**

- `TypeError`
  - The user has specified a keyword argument “study”.
- Other exceptions are raised by `yields.ccsne.fractional`.

### **See Also**

[yields.ccsne.fractional](#)

### **Example**

```
>>> from vice.yields.ccsne import WW95
>>> WW95.set_params(m_lower = 0.08, m_upper = 40, IMF = “salpeter”)
```

[Back to WW95](#)

### **References**

Woosley & Weaver (1995), ApJ, 101, 181

---

## **vice.yields.snea**

**Type Ia Supernovae Nucleosynthetic Yield Tools:** Here user's can calculate nucleosynthetic yields from type Ia supernovae (both single detonation and IMF-integrated) as well as modify their yield settings, which VICE will adopt in their simulations. VICE has built-in tables allowing users to calculate IMF-integrated yields from the results of supernovae simulations ran by two different studies, whih can also be directly imported as the yield settings.

### **Included Features**

- `fractional` (Type: `<function>`)  
Calculate an IMF-integrated yield of a given element.
- `settings` (Type: `dataframe`)  
Stores the user's current settings for these yields.
- `single` (Type: `<function>`)  
Look up the mass yield of a given element from a single type Ia supernova from a given study.

### **Built-in Yield Tables Available for Import**

- [seitenzahl13](#): Seitenzahl et al. (2013)
- [iwamoto99](#): Iwamoto et al. (1999)

### **References**

Iwamoto et al. (1999), ApJ, 124, 439  
Seitenzahl et al. (2013), MNRAS, 429, 1156

---

## **vice.yields.snea.fractional**

Calculate an IMF-integrated fractional nucleosynthetic yield of a given element from type Ia supernovae. Unlike `vice.yields.ccsne.fractional`, this function does not require numerical quadrature. See section 5.2 of VICE’s science documentation at <https://github.com/giganano/VICE/tree/master/docs> for further details.

**Signature:** `vice.yields.snea.fractional(element, study = “seitenzahl13”, model = “N1”, n = 2.2×10-3)`

### **Parameters**

- `element` (Type: str [case-insensitive])  
The symbol of the element to calculate the yield for.
- `study` (Type: str [case-insensitive]; Default: “seitenzahl13”)  
A keyword denoting which study to adopt the yield from.

### **Keywords and their Associated Studies**

- “seitenzahl13”: Seitzzahl et al. (2013), MNRAS, 429, 1156
- “iwamoto99”: Iwamoto et al. (1999), ApJ, 124, 439
- `model` (Type: str [case-insensitive]; Default: “N1”)  
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` (Type: real number; Default:  $2.2 \times 10^{-3}$ )  
The average number of type Ia supernovae produced per unit stellar mass formed  $N_{\text{Ia}}/M_*$ . This parameter has units  $M_{\odot}^{-1}$ . The default value is derived from Maoz & Mannucci (2012).

### **Returns**

- `yield` (Type: real number)  
The IMF-integrated yield. Unlike `vice.yields.ccsne.fractional`, there is no associated numerical error with this function, because the solution is analytic.

### **Raises**

- `ValueError`
  - The element is not built into VICE.
  - The study is not built into VICE.
  - $n < 0$
- `LookupError`
  - The model is not recognized for the given study.
- `IOError` [Occurs only if VICE’s file structure has been tampered with]
  - The parameters passed to this function are allowed but the data file is not found.

---

### Notes

This function evaluates the solution to the following equation:

$$y_x^{\text{Ia}} = \frac{N_{\text{Ia}}}{M_*} M_x \quad (3)$$

where  $M_x$  is the value returned by `vice.yields.snea.single`.

### Example

```
>>> vice.yields.snea.fractional("fe")
0.0025825957080000002
>>> vice.yields.snea.fractional("ca")
8.935489894764334e-06
>>> vice.yields.snea.fractional("ni")
0.00016576890932800003
```

### References

Maoz & Mannucci (2012), PASA, 29, 447

---

## **vice.yields.snea.settings**

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. 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**

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

### **vice.yields.snea.settings.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.yields.snea.settings.save_defaults()` immediately following this function.

### **vice.yields.snea.settings.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.yields.snea.settings.save_defaults()` immediately following this function.

### **vice.yields.snea.settings.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.

---

## **vice.yields.snea.single**

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. See section 5.2 of VICE's science documentation at <https://github.com/giganano/VICE/tree/master/docs> for further details.

**Signature:** `vice.yields.snea.single(element, study = "seitenzahl13", model = "N1")`

### **Parameters**

- **element** (Type: `str` [case-insensitive])  
The symbol of the element to look up the yield for.
- **study** (Type: `str` [case-insensitive]; Default: `"seitenzahl13"`)  
A keyword 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** (Type: `str` [case-insensitive]; Default: `"N1"`)  
A keyword denoting 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

### **Returns**

- **yield** (Type: `real number`)  
The mass yield of the given element in solar masses as determined for the specified model from the specified study.

### **Raises**

- **ValueError**
  - The element is not built into VICE.
  - The study is not built into VICE.
- **LookupError**
  - The study is recognized, but the model is not recognized for that particular study.
- **IOError** [Only occurs if VICE's internal file structure has been tampered with]
  - The data file is not found.

### **Notes**

The only calculation performed by this function is a sum of the mass yields of all isotopes of the given element reported by the study. Other than that, it is a simple lookup function through the file tree built into VICE.

---

**Example**

```
>>> vice.yields.snea.single("fe")
1.17390714
>>> vice.yields.snea.single("fe", study model = "W70")
0.77516
>>> vice.yields.snea.single("ni", model = "N100L")
0.03914090000000526
```

---

## **vice.yields.sneia.iwamoto99**

**Iwamoto et al. (1999), ApJ, 124, 439 Nucleosynthetic Yield Tools:** Importing this module will automatically set all yield settings from type Ia supernovae to the IMF-integrated yields as determined from the simulations ran by Iwamoto et al. (1999). It will default to the W70 explosion model.

VICE achieves this by calling `yields.sneia.fractional` for every element built into the software and storing the returned value in `yields.sneia.settings`.

[`iwamoto99.set\_params`](#): Update the parameters with which the yields are calculated.

### **Notes**

By importing this module, the user does not sacrifice the flexibility of VICE's user-specified yields. After importing this module, the fields of `vice.yields.sneia.settings` can still be modified in whatever manner the user sees fit.

This module is not imported with the simple `import vice` statement.

### **Example**

```
>>> from vice.yields.sneia import iwamoto99
>>> iwamoto99.set_params(n = 1.5e-03)
```



---

## **vice.yields.snea.iwamoto99.set\_params**

Update the parameters with which the yields are calculated from the Iwamoto et al. (1999) data.

### **Parameters**

- `kwargs` (varying types)  
Keyword arguments to pass to `yields.snea.fractional`.

### **Raises**

- `TypeError`
  - The user has specified a keyword argument “study”
- Other exceptions are raised by `yields.snea.fractional`.

### **See Also**

[yields.snea.fractional](#)

### **Example**

```
>>> from vice.yields.snea import iwamoto99
>>> iwamoto99.set_params(n = 1.5e-09)
```

[Back to iwamoto99](#)

### **References**

Iwamoto et al. (1999), ApJ, 124, 439

---

## **vice.yields.snea.seitenzahl13**

**Seitenzahl et al. (2013), MNRAS, 429, 1156 Nucleosynthetic Yield Tools:** Importing this module will automatically set all yield settings from type Ia supernovae to the IMF-integrated yields as determined from the simulations ran by Seitenzahl et al. (2013). It will default to the N1 explosion model.

VICE achieves this by calling `yields.snea.fractional` for every element built into the software and storing the returned value in `yields.snea.settings`.

[seitenzahl13.set\\_params](#): Update the parameters with which the yields are calculated.

### **Notes**

By importing this module, the user does not sacrifice the flexibility of VICE's user-specified yields. After importing this module, the fields of `vice.yields.snea.settings` can still be modified in whatever manner the user sees fit.

This module is not imported with the simple `import vice` statement.

### **Example**

```
>>> from vice.yields.snea import seitenzahl13
>>> seitenzahl13.set_params(n = 1.5e-03)
```

---

## **vice.yields.snea.seitenzahl13.set\_params**

Update the parameters with which the yields are calculated from the Seitenzahl et al. (2013) data.

### **Parameters**

- `kwargs` (varying types) Keyword arguments to pass to `yields.snea.fractional`.

### **Raises**

- `TypeError`
  - The user has specified a keyword argument “study”.
- Other exceptions are raised by `yields.snea.fractional`.

### **See Also**

[yields.snea.fractional](#)

### **Example**

```
>>> from vice.yields.snea import seitenzahl13
>>> seitenzahl13.set_params(n = 1.5e-03)
```

[Back to seitenzahl13](#)

### **References**

Seitenzahl et al. (2013), ApJ, 124, 439

---

## **vice.cumulative\_return\_fraction**

Determine the cumulative return fraction for a single stellar population at a given age. This quantity represents the fraction of the stellar population's mass that is returned to the ISM as gas at the birth metallicity of the stars. See section 2.2 VICE's science documentation for further details.

**Signature:** `vice.cumulative_return_fraction(age, IMF = "kroupa", m_upper = 100, m_lower = 0.08)`

### **Parameters**

- **age** (Type: real number)  
The age of the stellar population in Gyr.
- **IMF** (Type: str [case-insensitive]; Default: "kroupa")  
The stellar initial mass function (IMF) to assume. This must be either "kroupa" (1) or "salpeter" (2).
- **m\_upper** (Type: real number; Default: 100)  
The upper mass limit on star formation in solar masses.
- **m\_lower** (Type: real number; Default: 0.08)  
The lower mass limit on star formation in solar masses.

### **Returns**

- **crf** (Type: real number)  
The value of the cumulative return fraction for a stellar population at the specified age under the specified parameters.

### **Raises**

- **TypeError**
  - age is not a real number
  - IMF is not of type str
  - m\_upper is not a real number
  - m\_lower is not a real number
- **ValueError**
  - age < 0
  - IMF is not recognized
  - m\_upper ≤ 0
  - m\_lower ≤ 0
  - m\_lower ≥ m\_upper

---

### Notes

VICE operates under the approximation that stars have a mass-luminosity relationship of  $L \sim M^{4.5}$ , leading to a mass-lifetime relation that is also a power law of  $\tau \sim M/L \sim M^{-3.5}$ .

VICE implements the remnant mass model of Kalirai et al. (2008), assuming that (on average) stars above  $8 M_{\odot}$  leave behind remnants of  $1.44 M_{\odot}$ , while stars below  $8 M_{\odot}$  leave behind remnants of mass  $0.394 M_{\odot} + 0.019 M$ .

### Example

```
>>> vice.cumulative_return_fraction(1)
0.3560160079575864
>>> vice.cumulative_return_fraction(2)
0.38056657042902253
>>> vice.cumulative_return_fraction(3)
0.394760119115021
```

### References

Kalirai et al. (2008), ApJ, 676, 594  
(1) Kroupa (2001), MNRAS, 231, 322  
(2) Salpeter (1955), ApJ, 121, 161

---

## vice.history

Read in the part of a simulation's output that records the time-evolution of the ISM metallicity.

**Signature:** `vice.history(name)`

### Parameters

- `name` (Type: `str`)

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

### Returns

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

A VICE history object (a subclass of the `VICE dataframe`), which contains the time in Gyr, gas and stellar mass in solar masses, star formation and infall rates in  $M_{\text{sun}}/\text{yr}$ , inflow and outflow metallicities for each element, gas-phase mass and metallicities of each element, and every  $[X/Y]$  combination of abundance ratios for each output timestep.

### Raises

- `IOError` [Only occurs if the output has been tampered with]
  - The output file is not found.
  - The output file is not formatted correctly.
  - Other VICE output files are missing from the output.

### Notes

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, user's may analyze the results of VICE simulations in languages other than `python`.

In addition to the abundance and dynamical evolution information, `history` objects will also record the effective recycling parameter and the specified mass loading parameter at all output times. These are the *actual* recycling rate divided by the star formation rate and the instantaneous mass loading  $\eta(t)$  that the user has specified regardless of the smoothing time, respectively.

In addition to the keys present in `history dataframes`, users may also index them with `"z"` and `"[m/h]"` [case-insensitive]. They will determine the total metallicity by mass as well as the logarithmic abundance relative to solar. Both are scaled in the following manner:

$$Z = Z_{\odot} \frac{\sum_i Z_i}{\sum_i Z_i^{\odot}} \quad (4)$$

This is the scaling of the total metallicity that is encoded into VICE's timestep integrator, which prevents the simulation from behaving as if it has a systematically low metallicity when enrichment is tracked for only a small number of elements. See section 5.4 of VICE's science documentation at <https://github.com/giganano/VICE/tree/master/docs> for further details.

---

**Example**

```
>>> 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 end of simulation: %.2e" % (hist["[o/fe)"][-1]))
[O/Fe] at end of simulation: -3.12e-01
```

---

## vice.main\_sequence\_mass\_fraction

Determine the main sequence mass fraction for a single stellar population at a given age. This quantity represents the fraction of the stellar population's mass that is still in the form of stars on the main sequence. See section 2.3 of VICE's science documentation for further details.

**Signature:** `vice.main_sequence_mass_fraction(age, IMF = "kroupa", m_upper = 100, m_lower = 0.08)`

### Parameters

- `age` (Type: real number)  
The age of the stellar population in Gyr.
- `IMF` (Type: str [case-insensitive]; Default: "kroupa")  
The stellar initial mass function (IMF) to assume. This must be either "kroupa" (1) or "salpeter" (2).
- `m_upper` (Type: real number; Default: 100)  
The upper mass limit on star formation in solar masses.
- `m_lower` (Type: real number; Default: 0.08)  
The lower mass limit on star formation in solar masses.

### Returns

- `msmf` (Type: real number)  
The value of the main sequence mass fraction for a stellar population at the specified age under the specified parameters.

### Raises

- `TypeError`
  - `age` is not a real number
  - `IMF` is not of type `str`
  - `m_upper` is not a real number
  - `m_lower` is not a real number
- `ValueError`
  - `age` < 0
  - `IMF` is not recognized
  - `m_upper` ≤ 0
  - `m_lower` ≤ 0
  - `m_lower` ≥ `m_upper`

### Notes

VICE operates under the approximation that stars have a mass-luminosity relationship of  $L \sim M^{4.5}$ , leading to a mass-lifetime relation that is also a power law of  $\tau \sim M/L \sim M^{-3.5}$ .



---

### **Example**

```
>>> vice.main_sequence_mass_fraction(1)
0.5815004968281556
>>> vice.main_sequence_mass_fraction(2)
0.5445877675278488
>>> vice.main_sequence_mass_fraction(3)
0.5219564300200146
```

### **References**

- (1) Kroupa (2001), MNRAS, 231, 322
- (2) Salpeter (1955), ApJ, 121, 161

---

## vice.mdf

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

**Signature:** `vice.mdf(name)`

### Parameters

- `name` (Type: `str`)  
The name of the simulation output to read from, with or without the `‘.vice’` extension.

### Returns

- `zdist` (Type: `VICE dataframe`)  
A `VICE dataframe` containing the bin edges and the values of the normalized stellar metallicity distribution in each `[X/H]` abundance and `[X/Y]` abundance ratio.

### Raises

- `IOError` [Occurs only if the output has been tampered with]
  - The output file is not found.
  - The output file is not formatted correctly.
  - Other VICE output files are missing from the output.

### Notes

For an output under a given name, this file will be stored at `name.vice/mdf.out`, and it is a simple ascii text file with a comment header detailing each column. By storing the output in this manner, I allow user’s to analyze the results of VICE simulations in languages other than `python`.

VICE normalizes stellar metallicity distribution functions such that the area under the user-specified binspace is equal to 1. Because of this, they should be interpreted as probability densities. See section 6 of VICE’s science documentation at <https://github.com/giganano/VICE/tree/master/docs> for further details.

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.

Because the user-specified bins that the stellar MDF is sorted into may not be symmetric, if the simulation tracks the abundance ratios of stars in `[X/Y]`, the returned `dataframe` will *not* determine the distribution in the inverse abundance ratio `[Y/X]` automatically.

### Example

```
>>> 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
>>> [zdist[65]["bin_edge_left"], zdist[65]["bin_edge_right"]]
[2.50e-01, 3.00e-01]
```

---

## 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)`

### Parameters

- `output_obj` (Type: `vice.output`)  
Any `vice.output` object.

### Returns

- `sz` (Type: `vice.singlezone`)  
A `vice.singlezone` object with the same attributes as that which produced the given output.

### Raises

- `ImportError`
  - The output has encoded functional attributes and the user does not have `dill` installed.
- `UserWarning`
  - The output was produced with functional attributes, but was ran on a system without `dill`, and they have thus been lost.

### Notes

VICE stores attributes of `singlezone` objects in a pickle within the output directory. Encoding functions along with the rest of the attributes requires the package `dill`, an extension to pickle which makes this possible. If `dill` is not installed, these attributes will not be encoded with the output.

It is recommended that users install `dill` in order to make use of these features. It is installable via `'pip install dill'`.

### Example

```
>>> out = vice.output("example")
>>> new = vice.mirror(out)
>>> new
vice.singlezone{
  name -----> onezonemodel
  func -----> <function _DEFAULT_FUNC_ at 0x1085a6ae8>
  mode -----> ifr
  verbose -----> False
  elements -----> ('fe', 'sr', 'o')
  IMF -----> kroupa
  eta -----> 2.5
  enhancement ----> 1.0
  Zin -----> 0.0
  recycling -----> continuous
  delay -----> 0.15
  RIa -----> plaw
```

---

```
Mg0 -----> 6000000000.0
smoothing -----> 0.0
tau_ia -----> 1.5
tau_star -----> 2.0
schmidt -----> False
schmidt_index --> 0.5
MgSchmidt -----> 6000000000.0
dt -----> 0.01
m_upper -----> 100.0
m_lower -----> 0.08
Z_solar -----> 0.014
agb_model -----> cristallo11
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))
```

---

## vice.output

Reads in the output from the `singlezone` class and allows the user to access it easily via VICE dataframes. The results are read in automatically.

**Signature:** `vice.output.__init__(name)`

### Notes

Reinstancing functional attributes from VICE outputs requires the package `dill`, an extension to `pickle` in the python standard library. It is recommended that VICE users install `dill` if they have not already so that they can make use of this feature; this can be done via `pip install dill`.

VICE outputs are stored in directories with a “.vice” extension following the name of the simulation assigned to the `singlezone` object that produced it. Because the history and mdf outputs are stored in pure ascii text, this allows users to open them in languages other than python while retaining the ability to run `<command> *.vice` in a linux terminal to run operations on all VICE outputs in a given directory.

### See Also

- Function [history](#)
- Function [mdf](#)

---

## **Attributes**

- **name** (Type: str)  
The name of the “.vice” directory containing the output of a `singlezone` object. The “.vice” extension need not be specified with the name.
- **elements** (Type: tuple)  
The symbols for the elements whose enrichment was modeled to produce the output file.
- **history** (Type: dataframe)  
The dataframe read in via `vice.history(name)`.

### **See Also**

- Function [history](#)

- **mdf** (Type: dataframe)  
The dataframe read in via `vice.mdf(name)`.

### **See Also**

- Function [mdf](#)

- **ccsne\_yields** (Type: VICE dataframe)  
A dataframe encoding the settings for nucleosynthetic yields from core collapse supernovae at the time the simulation was ran.

### **See Also**

- [vice.yields.ccsne.settings](#)

- **sneia\_yields** (Type: VICE dataframe)  
A dataframe containing the yield settings from type Ia supernovae at the time the simulation was ran.

### **See Also**

- [vice.yields.sneia.settings](#)

## **Example**

```
>>> out = vice.output("example")
>>> sfr = out.history["sfr"]
>>> ofe = out.history["[O/Fe]"]
>>> hundredth_line = out.history[100]
```

---

## vice.output.show()

Show a plot of the given quantity referenced by a keyword argument [case-insensitive].

**Signature:** `vice.output.show(key, xlim = None, ylim = None)`

### Parameters

- `key` (Type: `str` [case-insensitive])

The keyword argument. If this is a quantity stored in the `history` attribute, it will be plotted against time by default. Conversely, if it is stored in the `mdf` attribute, 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.

- `xlim` (Type: array-like; Default: `None`)  
The x-limits to impose on the shown plot.
- `ylim` (Type: array-like; Default: `None`)  
The y-limits to impose on the shown plot.

### Raises

- `KeyError`
  - `key` is not found in either `history` or `mdf` attributes
- `ImportError/ModuleNotFoundError`
  - `matplotlib` version  $\geq 2$  is not found in the user's system.  
(The `ModuleNotFoundError` is raised in python versions  $\geq 3.6$ .)
- All other errors are raised by `matplotlib.pyplot.show`

### Notes

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.

### Example

```
>>> 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]")
```

---

## 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. See section 2.4 of VICE's science documentation for further details.

**Signature:** `vice.single_stellar_population(element, mstar = 1.e6, Z = 0.014, time = 10, dt = 0.01, m_upper = 100, m_lower = 0.08, IMF = "kroupa", RIa = "plaw", delay = 0.15, agb_model = "cristallo11")`

### Parameters

- **element** (Type: str [case-insensitive])  
The symbol of the element to simulate the enrichment for.
- **mstar** (Type: real number; Default:  $10^6$ )  
The birth mass of the star cluster in solar masses.
- **Z** (Type: real number; Default: 0.014)  
The metallicity by mass of the stars in the cluster. (i.e.  $Z = \text{mass of metals} / \text{total mass}$ )
- **time** (Type: real number; Default: 10)  
The amount of time in Gyr to run the simulation for.
- **dt** (Type: real number; Default: 0.01)  
The size of each timestep in Gyr.
- **m\_upper** (Type: real number; Default: 100)  
The upper mass limit on star formation in solar masses.
- **m\_lower** (Type: real number; Default: 0.08)  
The lower mass limit on star formation in solar masses.
- **IMF** (Type: str [case-insensitive]; Default: "kroupa")  
The stellar initial mass function to assume. This must be either "kroupa" for the Kroupa (2001), MNRAS, 322, 231 IMF or "salpeter" for the Salpeter (1955), ApJ, 121, 151 IMF.
- **RIa** (Type: str [case-insensitive] or <function>; Default: "plaw")  
The delay-time distribution  $R_{\text{Ia}}(t)$  to adopt. VICE will automatically normalize any function that is passed. Alternatively, VICE has built-in distributions: "plaw" (power-law,  $\propto t^{-1.1}$ ) and "exp" (exponential,  $\propto e^{-t/1.5 \text{ Gyr}}$ ).
- **delay** (Type: real number; Default: 0.15)  
The minimum delay time following the formation of a single stellar population before the onset of type Ia supernovae in Gyr.
- **agb\_model** (Type: str [case-insensitive]; Default: "cristallo11")  
A keyword 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

- `mass` (Type: list)  
The net mass of the element in solar produced by the star cluster at each timestep.
- `times` (Type: list)  
The times in Gyr corresponding to each mass yield.

## Raises

- `ValueError`
  - The element is not built into VICE.
  - `mstar < 0`
  - `Z < 0`
  - `time < 0` or `time > 15` [VICE does not simulate enrichment on timescales longer than the age of the universe]
  - `dt < 0`
  - `m_upper < 0`
  - `m_lower < 0`
  - `m_lower > m_upper`
  - The IMF is not built into VICE
  - `delay < 0`
  - `agb_model` is not built into VICE
- `LookupError`
  - `agb_model == "karakas10"` and the atomic number of the element is greater than or equal to 29. The Karakas (2010), MNRAS, 403, 1413 study did not report yields for elements heavier than nickel.
- `ArithmeticError`
  - A functional  $R_{\text{Ia}}$  evaluated to a negative value, inf, or NaN at any given timestep.
- `IOError` [Only occurs if VICE's file structure has been tampered with]
  - The AGB yield file was not found.

## Example

```
>>> mass, times = vice.single_stellar_population("sr", mstar = 1e6, Z = 0.008)
>>> mass[-1]
0.04808964406448721
>>> mass, times = vice.single_stellar_population("fe")
2679.816051685778
```

---

## vice.singlezone

Runs simulations of chemical enrichment under the single-zone approximation for user-specified parameters. The organizational structure of this class is simple; every attribute encodes information on a relevant galaxy evolution parameter.

**Signature:** `vice.singlezone.__init__(name = "onezonemodel", func = _DEFAULT_FUNC, mode = "ifr", elements = ["fe", "sr", "o"], imf = "kroupa", eta = 2.5, enhancement = 1, Zin = 0, recycling = "continuous", bins = _DEFAULT_FUNC, delay = 0.15, ria = "plaw", Mg0 = 6.0e+09, 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")`

**See Also** [<https://github.com/giganano/VICE/tree/master/docs>]

- Sections 3 - 6 of science documentation
- User's notes on [functional attributes](#) and [numerical delta functions](#)

---

## Attributes

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

The name of the simulation. 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.

## Notes

The user need not interact with any of the output files; the output object is designed to read in all of the results automatically.

Most of the relevant physical information stored in VICE outputs are in the `history.out` and `mdf.out` output files. They are simple ascii text files, allowing users to open them in languages other than python if they so choose. The other output files store the yield settings at the time of simulation and the integrator parameters which produced it.

By forcing a `".vice"` extension on the output file, users can run `<command> *.vice` in a linux terminal to run commands over all vice outputs in a given directory.

[Back to singlezone](#)

- **func** (Type: `<function>`; Default: `_DEFAULT_FUNC_`)

A callable python function of time which returns a real number. This must take only one parameter, which will be interpreted as time in Gyr. The value returned by this function will represent either the gas infall history in `Msun/yr`, the star formation history in `Msun/yr`, or the gas supply in `Msun`.

## Notes

The default function returns the value of 9.1 always. With a default `moe` of `"ifr"`, if these attributes are not changed, the simulation will run with an infall rate of 9.1 `Msun/yr`.

Encoding this functional attribute into VICE outputs requires the package `dill`, an extension to the python standard library. Without this, the outputs will not have memory of this parameter. It is recommended that VICE user install `dill` if they have not already so that they can make use of this feature; this can be done via `pip install dill`.

**See Also** [<https://github.com/giganano/VICE/tree/master/docs>]

- Section 3 of science documentation
- Attribute [mode](#).
- User's Notes on [functional attributes](#) and [numerical delta functions](#).

[Back to singlezone](#)

- 
- **mode** (Type: `str` [case-insensitive]; Default: `"ifr"`)

The interpretation of the attribute `func`.

`mode = "ifr"`

The values returned from the attribute `func` represents the rate of gas infall into the galaxy in  $M_{\odot} \text{ yr}^{-1}$ .

`mode = "sfr"`

The value returned from the attribute `func` represents the star formation rate in  $M_{\odot} \text{ yr}^{-1}$ .

`mode = "gas"`

The values returned from the attribute `func` represent the mass of the ISM gas in  $M_{\odot}$ .

#### **Notes**

The attribute `func` will always be interpreted as taking time in Gyr as a parameter. However, infall and star formation rates will be interpreted as having units of  $M_{\odot}/\text{yr}$  according to convention.

#### **See Also**

- Section 3.1 of science documentation
- Attribute [func](#)

[Back to singlezone](#)

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

The symbols for the elements to track the enrichment for. In its current state, VICE recognizes all 76 astrophysically produced elements between carbon (`"c"`) and bismuth (`"bi"`).

#### **Notes**

The order in which the elements appear in this tuple will dictate the ratios that are quoted in the output stellar metallicity distribution function. For example, if element X appears before element Y, then VICE will determine the MDF in  $dN/d[Y/X]$ . Under this construction, users should put their “reference elements” first.

#### **See Also**

- Section 6 of science documentation

[Back to singlezone](#)

- 
- IMF (Type: str [case-insensitive]; Default: “kroupa”)

The assumed stellar initial mass function (IMF). This must be either “kroupa” (1) or “salpeter” (2). These IMFs have the following form:

“kroupa”

$$dN/dM \sim M^{-\alpha}$$

$$a = \begin{cases} 2.3 & M \geq M_{\odot} \\ 1.3 & 0.08M_{\odot} \leq M \leq 0.5M_{\odot} \\ 0.3 & M \leq M_{\odot} \end{cases} \quad (5)$$

“salpeter”

$$dN/dM \sim M^{-2.35} \quad (6)$$

### Notes

A future update to VICE will likely include functionality for a wider sample of IMFs.

### See Also

- The IMF is relevant in many sections of VICE’s science documentation.

### References

- (1) Kroupa (2001), MNRAS, 322, 231
- (2) Salpeter (1955), ApJ, 121, 161

[Back to singlezone](#)

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

The mass loading parameter: The ratio of the outflow rate to the star formation rate.

**Notes** If the smoothing timescale is nonzero, this relationship is more complicated. See associated docstring for further details.

If type <function>

Encoding this functional attribute into VICE outputs requires the package dill, an extension to pickle in the python standard library. Without this, the outputs will not have memory of this parameter. It is recommended that VICE users install dill if they have not already in order to make use of this feature; this can be done via `pip install dill`.

**See Also** [<https://github.com/giganano/VICE/tree/master/docs>]

- Section 3.2 of science documentation
- Attribute [smoothing](#)
- User’s Notes on [functional attributes](#) and [numerical delta functions](#).

[Back to singlezone](#)

- 
- **enhancement** (Type: real number of <function>; Default: 1.0)

The ratio of the outflow to ISM metallicities. This can also be a callable function of time in Gyr.

#### Notes

This multiplicative factor will apply to all elements tracked by the simulation.

If type <function>

Encoding this functional attribute into VICE outputs requires the package `dill`, an extension to `pickle` in the python standard library. Without this, the outputs will not have memory of this parameter. It is recommended that VICE users install `dill` if they have not already so that they can make use of this feature; this can be done via `'pip install dill'`.

#### See Also

- Sections 3.2 and 4.1 of science documentation
- Attribute [eta](#)
- Attribute [smoothing](#)

[Back to singlezone](#)

- **Zin** (Type: real number, <function>, or dataframe; Default: 0.0)

The metallicity of gas inflow. If this is a number or a function, it will apply to all elements tracked by the simulation. A python dictionary or VICE dataframe can be passed, allowing real numbers and functions to be assigned to each individual element.

#### Notes

The easiest way to switch this attribute to a dataframe is by passing an empty python dictionary (i.e. {}).

If type <function>

Encoding this functional attribute into VICE outputs requires the package `dill`, an extension to `pickle` in the python standard library. Without this, the outputs will not have memory of this parameter. It is recommended that VICE users install `dill` if they have not already so that they can make use of this feature; this can be done via `'pip install dill'`.

#### Example

```
>>> sz = vice.singlezone(name = "example")
>>> sz.Zin = {}
>>> sz.Zin
vice.dataframe{
  sr -----> 0.0
  fe -----> 0.0
  o -----> 0.0
}
>>> sz.Zin["fe"] = vice.solar_z["fe"]
>>> sz.Zin["o"] = lambda t: vice.solar_z["o"] * (t / 10.0)
>>> sz.Zin
vice.dataframe{
  sr -----> 0.0
  fe -----> 0.00129
  o -----> <function <lambda> at 0x11559aa0>
}
```

[Back to singlezone](#)

- 
- **recycling** (Type: str [case-insensitive] or real number; Default: “continuous”)

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.

If this attribute is a string, it must be “continuous” [case-insensitive]. In this case VICE will treat recycling from each episode of star formation individually via a treatment of the stellar initial mass function and the remnant mass model of Kalirai et al. (2008).

If this attribute is a real number, it must be a value between 0 and 1. VICE will implement instantaneous recycling in this case, and this parameter will represent the fraction of a single stellar population’s mass that is returned instantaneously the ISM.

#### **Notes**

It is recommended that user’s adopt  $r = 0.4$  (0.2) if they desire instantaneous recycling with a Kroupa (1) (Salpeter (2)) IMF, based on the analytical model of Weinberg, Andrews & Freudenburg (2017).

**See Also** [<https://github.com/giganano/VICE/tree/master/docs>]

- Section 3.3 of science documentation

#### **Example**

```
>>> sz = vice.singlezone(name = “example”)
>>> sz.recycling = 0.4
>>> sz.IMF = “salpeter”
>>> sz.recycling = 0.2
```

#### **References**

Kalirai et al. (2008), ApJ, 676, 594

(1) Kroupa (2001), MNRAS, 322, 231

(2) Salpeter (1955), ApJ, 131, 161

Weinberg, Andrews & Freudenburg (2017), ApJ, 837, 183

[Back to singlezone](#)

- **bins** (Type: array-like [elements are real numbers]; Default [-3, -2.95, -2.9, ... , 0.9, 0.95, 1.0])

The bins in each [X/H] abundance and [X/Y] abundance ratio to sort the normalized stellar metallicity distribution function into. By default, VICE sorts everything into 0.05-dex width bins between [X/H] and [X/Y] = -3 and +1.

#### **Notes**

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

**See Also** [<https://github.com/giganano/VICE/tree/master/docs>]

- Section 6 of science documentation

[Back to singlezone](#)

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

The minimum delay time in Gyr for the onset of type Ia supernovae associated with a single stellar population. The default parameter is adopted from Weinberg, Andrews & Freudenburg (2017).

**See Also** [<https://github.com/giganano/VICE/tree/master/docs>]

- Attribute [ria](#)
- Section 4.3 of science documentation

### **References**

Weinberg, Andrews & Freudenburg (2017), ApJ, 837, 183

- **RIa** (Type: str [case-insensitive] or <function>; Default: “plaw”)

The delay-time distribution (DTD) for type Ia supernovae to adopt. If type str, VICE will use built-in DTDs:

- **“exp”**  
 $R_{\text{Ia}} \sim e^{-t}$  [e-folding timescale set by attribute [tau\\_ia](#)]
- **“plaw”**  
 $R_{\text{Ia}} \sim t^{-1.1}$

Alternatively, the user may pass their own function of time in Gyr, and the normalization of the custom DTD will be taken care of automatically.

If type <function>

Encoding this functional attribute into VICE outputs requires the package `dill`, an extension to `pickle` in the python standard library. Without this, the outputs will not have memory of this parameter. It is recommended that VICE users install `dill` if they have not already in order to make use of this feature; this can be done via `pip install dill`.

**See Also** [<https://github.com/giganano/VICE/tree/master/docs>]

- Section 4.3 of science documentation
- User’s note on [functional attributes](#) and [numerical delta functions](#)

[Back to singlezone](#)



- 
- **Mg0** (Type: real number; Default:  $6.0 \times 10^9$ )

The mass of the ISM gas at time  $t = 0$  in  $M_\odot$ .

#### Notes

This parameter only matters when the simulation is in infall mode (i.e. `mode = "ifr"`). In gas mode, `func(0)` specifies the initial gas supply, and in star formation mode, it is `func(0) * tau_star(0)` (modulo the prefactors imposed by gas-dependent star formation efficiency, if applicable).

[Back to singlezone](#)

- **smoothing** (Type: real number; Default: 0.0)

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 (attribute [eta](#)). For an outflow rate  $\dot{M}_{\text{out}}$  and star formation rate  $\dot{M}_*$  with smoothing time  $\tau$ :

$$\dot{M}_{\text{out}} = \langle \dot{M}_* \rangle_\tau \quad (7)$$

#### Notes

While this parameter time-averages the star formation rate, it does NOT time-average the mass-loading factor.

**See Also** [<https://github.com/giganano/VICE/tree/master/docs>]

- Section 3.2 of science documentation

[Back to singlezone](#)

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

The e-folding timescale in Gyr of an exponentially decaying delay-time distribution for type Ia supernovae.

#### Notes

Because this is an e-folding timescale, it only matters when attribute `ria` = “exp”.

**See Also** [<https://github.com/giganano/VICE/tree/master/docs>]

- Section 4.3 of science documentation

[Back to singlezone](#)

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

The star formation rate per unit gas supply in Gyr ( $M_{\text{gas}}/\dot{M}_*$ ). In observational journal articles, this is sometimes 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.

#### Notes

When attribute `schmidt` = True, this is interpreted as the prefactor on gas-dependent star formation efficiency.

If type <function>

Encoding this functional attribute into VICE outputs requires the package `dill`, an extension to `pickle` in the python standard library. Without this, the outputs will not have memory of this parameter. It is recommended that VICE users install `dill` if they have not already so that they can make use of this feature; this can be done via `'pip install dill'`.

**See Also** [<https://github.com/giganano/VICE/tree/master/docs>]

- Section 3.1 of science documentation

[Back to singlezone](#)

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

The timestep size in Gyr to use in the integration.

#### **Notes**

For fine timesteps with a given ending time in the simulation, this affects the total integration time with a  $\Delta t^{-2}$  dependence.

- `Schmidt` (Type: bool; Default: False)

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 attributes `tau_star`, `MgSchmidt`, and `schmidt_index` determine the star formation efficiency at that timestep via:

$$\tau_*^{-1} = \text{tau\_star}(t)^{-1} \left( \frac{M_{\text{gas}}}{M_{\text{gSchmidt}}} \right)^{\text{schmidt\_index}} \quad (8)$$

**See Also** [<https://github.com/giganano/VICE/tree/master/docs>]

- Section 3.1 of science documentation

#### **References**

Leroy et al. (2008), AJ, 136, 2782

Schmidt (1959), ApJ, 129, 243

[Back to singlezone](#)

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

The power-law index on gas-dependent star formation efficiency.

**See Also** [<https://github.com/giganano/VICE/tree/master/docs>]

- Section 3.1 of science documentation
- Attribute `schmidt`

[Back to singlezone](#)

- `MgSchmidt` (Type: real number; Default:  $6.0 \times 10^9$ )

The normalization of the gas supply when star formation efficiency is dependent on the gas supply. **Notes**

In practice, 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 Also** [<https://github.com/giganano/VICE/tree/master/docs>]

- Section 3.1 of science documentation
- Attribute `schmidt`

[Back to singlezone](#)

- 
- `m_upper` (Type: real number; Default: 100)  
The upper mass limit on star formation in  $M_{\odot}$ .

[Back to singlezone](#)

- `m_lower` (Type: real number; Default: 0.08)  
The lower mass limit on star formation in  $M_{\odot}$ .

[Back to singlezone](#)

- 
- **Z\_solar** (Type: real number; Default: 0.014 (Asplund et al. 2009))

The metallicity by mass of the sun. This is used in calibrating the total metallicity of the ISM, which is necessary when there are only a few elements tracked by the simulation.

**See Also** [<https://github.com/giganano/VICE/tree/master/docs>]

- Section 5.4 of science documentation

### **References**

Asplund et al. (2009), ARA&A, 47, 481

[Back to singlezone](#)

- **agb\_model** (Type: str [case-insensitive]; Default: “crystallo11”)

A keyword denoting which stellar mass-metallicity grid of fractional nucleosynthetic yields from asymptotic giant branch stars to adopt in the simulation.

### **Recognized Keywords and their Associated Studies**

- “crystallo11”: Cristallo et al. (2011), ApJS, 197, 17
- “karakas10”: Karakas (2010), MNRAS, 403, 1413

### **Notes**

If the Karakas (2010) set of yields are adopted and any elements tracked by the simulation are heavier than nickel, a LookupError will be raised. The Karakas (2010) study did not report yields for elements heavier than nickel.

**See Also** [<https://github.com/giganano/VICE/tree/master/docs>]

- Sections 4.4 and 5.3 of science documentation

[Back to singlezone](#)

---

## **vice.singlezone.run()**

Runs the 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 an output object at any time.

**Signature:** `vice.singlezone.run(output_times, capture = False, overwrite = False)`

### **Parameters**

- `output_times` (Type: array-like [elements are real numbers])  
The times in Gyr at which VICE should record output from the simulation. These need not be sorted in any way; VICE will take care of that automatically.
- `capture` (Type: bool; Default: False)  
A boolean describing whether or not to return an output object from the results of the simulation.
- `overwrite` (Type: bool; Default: False)  
A boolean describing whether or not to force overwrite any existing files under the same name this simulation's output files.

### **Returns**

- `out` (Type: output [one returned in `capture = True`])  
An output object produced from this simulation's output.

### **Raises**

- `TypeError`
  - Any functional attribute evaluates to a non-numerical value at any timestep
- `ValueError`
  - Any element of `output_times` is negative
  - An inflow metallicity evaluates to a negative value
- `ArithmeticError`
  - An inflow metallicity evaluates to NaN or inf
  - Any functional attribute evaluates to NaN or inf at any timestep
- `UserWarning`
  - Any yield settings or class attributes are callable functions and the user does not have dill installed
- `ScienceWarning`
  - Any element tracked by the simulation is enriched in significant part by the r-process
  - Any element tracked by the simulation has a weakly constrained solar abundance measurement

---

### **Example**

```
>>> import numpy as np
>>> example = vice.singlezone(name = "example")
>>> outtimes = np.linspace(0, 10, 1001)
>>> print(outtimes[:10])
array([0., 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09])
>>> out = example.run(outtimes, capture = True)
>>> isinstance(out, vice.output)
True
```



---

## **Exception Handling**

VICE provides a unique warning class: `ScienceWarning`, which inherits from `UserWarning`. This is a warning class designed to treat as a distinct set of warnings those related to the scientific accuracy or precision of values returned from any given function. Although it is not recommended, users can silence this specific class of warnings via:

```
>>> warnings.filterwarnings("ignore", category = vice.ScienceWarning)
```

Alternatively, they may silence all warnings within VICE via

```
>>> vice.warnings.filterwarnings("ignore").
```

To silence all warnings globally:

```
>>> warnings.filterwarnings("ignore")
```

Other than the `ScienceWarning`, VICE does not provide any unique exception subclasses, and operates within the python standard library framework for exception handling.

---

## User's Notes

### 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 VICE 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 potentially not in the parameter which was meant to exhibit one.

When running a simulation, I 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.