

STRING_ARMA_2

1.0

Generated by Doxygen 1.8.17

1 Module Index	1
1.1 Modules	1
2 Class Index	3
2.1 Class List	3
3 Module Documentation	5
3.1 Non-member functions	5
3.1.1 Detailed Description	5
3.1.2 Function Documentation	5
3.1.2.1 calc_ivm()	5
4 Class Documentation	7
4.1 bypass_diode Class Reference	7
4.1.1 Detailed Description	8
4.1.2 Constructor & Destructor Documentation	8
4.1.2.1 bypass_diode() [1/2]	8
4.1.2.2 bypass_diode() [2/2]	9
4.1.3 Member Function Documentation	9
4.1.3.1 calcderv() [1/2]	9
4.1.3.2 calcderv() [2/2]	9
4.1.3.3 calcdiode() [1/2]	9
4.1.3.4 calcdiode() [2/2]	10
4.1.3.5 getldiode() [1/2]	10
4.1.3.6 getldiode() [2/2]	10
4.1.3.7 getlr() [1/2]	10
4.1.3.8 getlr() [2/2]	11
4.1.3.9 getTc() [1/2]	11
4.1.3.10 getTc() [2/2]	11
4.1.3.11 setldiode() [1/2]	11
4.1.3.12 setldiode() [2/2]	11
4.1.3.13 setlr() [1/2]	12
4.1.3.14 setlr() [2/2]	12
4.1.3.15 setTc() [1/2]	12
4.1.3.16 setTc() [2/2]	12
4.1.4 Member Data Documentation	12
4.1.4.1 ldiodi	13
4.1.4.2 lr	13
4.1.4.3 Tc	13
4.2 Classcomp Struct Reference	13
4.2.1 Detailed Description	13
4.2.2 Member Function Documentation	13
4.2.2.1 operator() [1/2]	14

4.2.2.2 operator() [2/2]	14
4.3 Gnuplot Class Reference	14
4.3.1 Constructor & Destructor Documentation	14
4.3.1.1 Gnuplot() [1/2]	14
4.3.1.2 ~Gnuplot() [1/2]	14
4.3.1.3 Gnuplot() [2/2]	15
4.3.1.4 ~Gnuplot() [2/2]	15
4.3.2 Member Function Documentation	15
4.3.2.1 operator() [1/2]	15
4.3.2.2 operator() [2/2]	15
4.3.3 Member Data Documentation	15
4.3.3.1 gnuplotpipe	15
4.4 grupString Struct Reference	15
4.4.1 Detailed Description	16
4.4.2 Member Data Documentation	16
4.4.2.1 lsc	16
4.4.2.2 Limit	16
4.4.2.3 N	16
4.4.2.4 SVbr	17
4.4.2.5 SVoc	17
4.4.2.6 SVocr	17
4.5 inMapCell Struct Reference	17
4.5.1 Detailed Description	18
4.5.2 Member Data Documentation	18
4.5.2.1 inMapDetails	18
4.5.2.2 inMapSum	18
4.6 inVectorCell Struct Reference	18
4.6.1 Detailed Description	19
4.6.2 Member Data Documentation	19
4.6.2.1 inVectorDetails	19
4.6.2.2 inVectorSum	19
4.7 solar_cell Class Reference	20
4.7.1 Detailed Description	22
4.7.2 Constructor & Destructor Documentation	23
4.7.2.1 solar_cell() [1/4]	23
4.7.2.2 solar_cell() [2/4]	23
4.7.2.3 solar_cell() [3/4]	24
4.7.2.4 solar_cell() [4/4]	24
4.7.3 Member Function Documentation	24
4.7.3.1 calcderi() [1/2]	24
4.7.3.2 calcderi() [2/2]	24
4.7.3.3 calcderv() [1/2]	25

4.7.3.4 calcderv() [2/2]	25
4.7.3.5 calcfcn() [1/2]	25
4.7.3.6 calcfcn() [2/2]	25
4.7.3.7 getG() [1/2]	26
4.7.3.8 getG() [2/2]	26
4.7.3.9 getlcell() [1/2]	26
4.7.3.10 getlcell() [2/2]	26
4.7.3.11 getIndex() [1/2]	26
4.7.3.12 getIndex() [2/2]	27
4.7.3.13 getlo() [1/2]	27
4.7.3.14 getlo() [2/2]	27
4.7.3.15 getlph() [1/2]	27
4.7.3.16 getlph() [2/2]	27
4.7.3.17 getlsc() [1/2]	28
4.7.3.18 getlsc() [2/2]	28
4.7.3.19 getTc() [1/2]	28
4.7.3.20 getTc() [2/2]	28
4.7.3.21 getVbreak() [1/2]	28
4.7.3.22 getVbreak() [2/2]	29
4.7.3.23 getVcell() [1/2]	29
4.7.3.24 getVcell() [2/2]	29
4.7.3.25 getVoc() [1/2]	29
4.7.3.26 getVoc() [2/2]	29
4.7.3.27 setG() [1/2]	30
4.7.3.28 setG() [2/2]	30
4.7.3.29 setlcell() [1/2]	30
4.7.3.30 setlcell() [2/2]	30
4.7.3.31 setIndex() [1/2]	30
4.7.3.32 setIndex() [2/2]	31
4.7.3.33 setlo() [1/2]	31
4.7.3.34 setlo() [2/2]	31
4.7.3.35 setlph() [1/2]	31
4.7.3.36 setlph() [2/2]	31
4.7.3.37 setlsc() [1/2]	32
4.7.3.38 setlsc() [2/2]	32
4.7.3.39 setTc() [1/2]	32
4.7.3.40 setTc() [2/2]	32
4.7.3.41 setVbreak() [1/2]	32
4.7.3.42 setVbreak() [2/2]	32
4.7.3.43 setVcell() [1/2]	33
4.7.3.44 setVcell() [2/2]	33
4.7.3.45 setVoc() [1/2]	33

4.7.3.46 setVoc() [2/2]	33
4.7.4 Member Data Documentation	33
4.7.4.1 G	34
4.7.4.2 lcell	34
4.7.4.3 index	34
4.7.4.4 lo	34
4.7.4.5 lph	34
4.7.4.6 lsc	34
4.7.4.7 Tc	35
4.7.4.8 Vbreak	35
4.7.4.9 Vcell	35
4.7.4.10 Voc	35
4.8 solar_string Class Reference	35
4.8.1 Detailed Description	38
4.8.2 Constructor & Destructor Documentation	38
4.8.2.1 solar_string() [1/2]	38
4.8.2.2 ~solar_string() [1/2]	38
4.8.2.3 solar_string() [2/2]	39
4.8.2.4 ~solar_string() [2/2]	39
4.8.3 Member Function Documentation	39
4.8.3.1 AccList() [1/2]	39
4.8.3.2 AccList() [2/2]	39
4.8.3.3 classifica() [1/2]	39
4.8.3.4 classifica() [2/2]	40
4.8.3.5 findValues() [1/2]	40
4.8.3.6 findValues() [2/2]	40
4.8.3.7 genList() [1/2]	40
4.8.3.8 genList() [2/2]	41
4.8.3.9 getambDiode() [1/2]	41
4.8.3.10 getambDiode() [2/2]	41
4.8.3.11 getIdiode() [1/2]	41
4.8.3.12 getIdiode() [2/2]	41
4.8.3.13 getlscmin() [1/2]	42
4.8.3.14 getlscmin() [2/2]	42
4.8.3.15 getSvbr() [1/2]	42
4.8.3.16 getSvbr() [2/2]	42
4.8.3.17 getSvcell() [1/2]	42
4.8.3.18 getSvcell() [2/2]	43
4.8.3.19 getSvoc() [1/2]	43
4.8.3.20 getSvoc() [2/2]	43
4.8.3.21 getVdiode() [1/2]	43
4.8.3.22 getVdiode() [2/2]	43

4.8.3.23	getVstring() [1/2]	44
4.8.3.24	getVstring() [2/2]	44
4.8.3.25	grups() [1/2]	44
4.8.3.26	grups() [2/2]	44
4.8.3.27	inici_corda() [1/2]	44
4.8.3.28	inici_corda() [2/2]	44
4.8.3.29	minim() [1/2]	45
4.8.3.30	minim() [2/2]	45
4.8.3.31	operator=() [1/2]	45
4.8.3.32	operator=() [2/2]	45
4.8.3.33	setBounds() [1/2]	46
4.8.3.34	setBounds() [2/2]	46
4.8.3.35	setIdiode() [1/2]	46
4.8.3.36	setIdiode() [2/2]	46
4.8.3.37	setSvbr() [1/2]	46
4.8.3.38	setSvbr() [2/2]	46
4.8.3.39	setSVbrx() [1/2]	47
4.8.3.40	setSVbrx() [2/2]	47
4.8.3.41	setSvcell() [1/2]	47
4.8.3.42	setSvcell() [2/2]	47
4.8.3.43	setSvoc() [1/2]	47
4.8.3.44	setSvoc() [2/2]	47
4.8.3.45	setVstring() [1/2]	48
4.8.3.46	setVstring() [2/2]	48
4.8.3.47	sort_string() [1/2]	48
4.8.3.48	sort_string() [2/2]	48
4.8.3.49	update_electrical() [1/2]	48
4.8.3.50	update_electrical() [2/2]	49
4.8.3.51	update_physical() [1/2]	49
4.8.3.52	update_physical() [2/2]	49
4.8.4	Member Data Documentation	49
4.8.4.1	CellsGr	50
4.8.4.2	corda	50
4.8.4.3	Idiode	50
4.8.4.4	midaCorda	50
4.8.4.5	rm3_c	51
4.8.4.6	Vdiode	51
4.9	solpan Struct Reference	51
4.9.1	Member Data Documentation	51
4.9.1.1	numStrings	51
4.9.1.2	panel	51
4.9.1.3	Vp	51

4.10 TableStr Struct Reference	52
4.10.1 Detailed Description	52
4.10.2 Member Data Documentation	52
4.10.2.1 index	52
4.10.2.2 IscGrup	52
4.10.2.3 SVbr	53
4.10.2.4 SVbrx	53
4.10.2.5 SVoc	53
Index	55

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Non-member functions	5
--------------------------------	---

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

bypass_diode	Represents a common component of a photovoltaic generator, a bypass diode	7
Classcomp	Comparison function object for the multimap	13
Gnuplot	14
grupString	Structure to gather global information of a group of cells that share, at least, the same short-circuit current	15
inMapCell	Structure with info of groups of cells with the same Isc and Vbrx	17
inVectorCell	Vector meant to contain info of groups of cells with the same short-circuit current (Isc)	18
solar_cell	Represents a PV cell, the most basic element of a solar generator	20
solar_string	Represents a string of solar cells group under the same bypass diode	35
solpan	51
TableStr	Defines the total parameters of a group of PV cells in the same string that have the same short-circuit current	52

Chapter 3

Module Documentation

3.1 Non-member functions

Functions

- double `calc_ivm` (`solar_string` *st, double Vp, int _dimX, int nS)
Calculates the exact state of the provided PV panel for a given voltage.

3.1.1 Detailed Description

3.1.2 Function Documentation

3.1.2.1 `calc_ivm()`

```
double calc_ivm (
    solar_string * st,
    double Vp,
    int _dimX,
    int nS )
```

Calculates the exact state of the provided PV panel for a given voltage.

The iterative method of Newton-Raphson is used to solve the nonlinear system of equations that define the PV panel. By default, it uses the euclidean norm. There is no maximum number of iterations defined.

Parameters

<i>*st</i>	Array of <code>solar_string</code> classes that conform the panel.
<i>Vp</i>	Voltage between the terminals of the PV panel [V].
<i>_dimX</i>	Sum of the total number of cells in the panel plus number of strings plus 1.
<i>nS</i>	Number of strings in the panel.

Returns

A double data type value of the total current through the panel [A].

Chapter 4

Class Documentation

4.1 bypass_diode Class Reference

Represents a common component of a photovoltaic generator, a bypass diode.

```
#include <bypass_diode.h>
```

Public Member Functions

- [bypass_diode](#) (void)
 - void [setTc](#) (double)
 - void [setIr](#) (void)
 - double [calcdiode](#) (double)
 - void [setldiode](#) (double)
 - double [getIr](#) (void)
 - double [getTc](#) (void)
 - double [getldiode](#) (void)
 - double [calcderv](#) (double)
- [bypass_diode](#) (void)
 - Constructor of the class [bypass_diode](#).*
- void [setTc](#) (double)
 - Set a double value for the Temperature of the diode [°C].*
- void [setIr](#) (void)
 - Updates the value for the reverse saturation current [A] according to the current value of the temperature of the diode Tc.*
- double [calcdiode](#) (double)
 - Calculates the fd function described in the [bypass_ch](#) part of the [mainPage](#).*
- void [setldiode](#) (double)
 - Updates the value for the current in the diode [A].*
- double [getIr](#) (void)
 - Gets the diode's reverse saturation current [A].*
- double [getTc](#) (void)
 - Gets the diode's temperature [°C].*
- double [getldiode](#) (void)
 - Gets the diode's current [A].*
- double [calcderv](#) (double)
 - Calculates the partial derivative respect the voltage of the diode, Vd, of the fd function described in the [math](#) part of the [mainPage](#).*

Protected Attributes

- double `Ir`
Reverse saturation current [A].
- double `Tc`
Current temperature of the bypass diode [°C].
- double `Idiode`
Current through the diode [A].

4.1.1 Detailed Description

Represents a common component of a photovoltaic generator, a bypass diode.

These diodes are placed in anti-parallel configuration with one or more cells. The cells grouped by the same bypass diode are considered strings. This class contains all the parameters that define a diode and to perform the calculations needed.

When the class is created, it contains the following reference values:

- **Irref** The reverse saturation current is 5e-6 A.
- **Tcref** The temperature of the cell equals 25.0 °C.

Other mathematical constants used in the calculations are:

- **k** Boltzmann constant: 1.38e-23 J/°K
- **q** Charge of an electron: 1.602e-19 C
- **m** Ideality factor of 1.5.

See also

[solar_cell](#)
[solar_string](#)

Note

The theoretical concepts behind this class are explained in the `bypass_ch` section of the mainPage.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 `bypass_diode()` [1/2]

```
bypass_diode::bypass_diode (  
    void )
```


4.1.2.2 bypass_diode() [2/2]

```
bypass_diode::bypass_diode (  
    void )
```

Constructor of the class [bypass_diode](#).

Uses all the reference values.

4.1.3 Member Function Documentation

4.1.3.1 calcderv() [1/2]

```
double bypass_diode::calcderv (  
    double Vd )
```

4.1.3.2 calcderv() [2/2]

```
double bypass_diode::calcderv (  
    double )
```

Calculates the partial derivative respect the voltage of the diode, V_d , of the fd function described in the math part of the mainPage.

Parameters

V_d	Double value of the diode's voltage V_d [V].
-------	------------------------------------------------

Returns

A double type with the value of the partial derivative respect the voltage of the diode of the funtion fd.

See also

[math](#)

4.1.3.3 calcIdiode() [1/2]

```
double bypass_diode::calcIdiode (  
    double Vdiode )
```

4.1.3.4 `calcIdiode()` [2/2]

```
double bypass_diode::calcIdiode (
    double )
```

Calculates the fd function described in the `bypass_ch` part of the `mainPage`.

Parameters

<i>Vdiode</i>	Double value of the diode's voltage Vd [V].
---------------	---------------------------------------------

Returns

A double type with the value of the funtion fd [A].

4.1.3.5 `getIdiode()` [1/2]

```
double bypass_diode::getIdiode (
    void )
```

4.1.3.6 `getIdiode()` [2/2]

```
double bypass_diode::getIdiode (
    void )
```

Gets the diode's current [A].

Returns

A double type with the value of the current [A].

4.1.3.7 `getIr()` [1/2]

```
double bypass_diode::getIr (
    void )
```

4.1.3.8 getIr() [2/2]

```
double bypass_diode::getIr (
    void )
```

Gets the diode's reverse saturation current [A].

Returns

A double type with the value of the reverse saturation current [A].

Warning

NOT IMPLEMENTED

4.1.3.9 getTc() [1/2]

```
double bypass_diode::getTc (
    void )
```

4.1.3.10 getTc() [2/2]

```
double bypass_diode::getTc (
    void )
```

Gets the diode's temperature [°C].

Returns

A double type with the value of the temperature [°C].

Warning

NOT IMPLEMENTED

4.1.3.11 setIdiode() [1/2]

```
void bypass_diode::setIdiode (
    double _Id )
```

4.1.3.12 setIdiode() [2/2]

```
void bypass_diode::setIdiode (
    double )
```

Updates the value for the current in the diode [A].

Parameters

↔	Double value for the diode's current [A].
↔ _↔ <i>Id</i>	

4.1.3.13 setIr() [1/2]

```
void bypass_diode::setIr (
    void )
```

4.1.3.14 setIr() [2/2]

```
void bypass_diode::setIr (
    void )
```

Updates the value for the reverse saturation current [A] according to the current value of the temperature of the diode Tc.

4.1.3.15 setTc() [1/2]

```
void bypass_diode::setTc (
    double _Tc )
```

4.1.3.16 setTc() [2/2]

```
void bypass_diode::setTc (
    double )
```

Set a double value for the Temperature of the diode [°C].

Parameters

_Tc	Double value for the temperature of the diode [°C].
-----	-----------------------------------------------------

4.1.4 Member Data Documentation

4.1.4.1 Idiode

```
double bypass_diode::Idiode [protected]
```

Current through the diode [A].

4.1.4.2 Ir

```
double bypass_diode::Ir [protected]
```

Reverse saturation current [A].

4.1.4.3 Tc

```
double bypass_diode::Tc [protected]
```

Current temperature of the bypass diode [°C].

4.2 Classcomp Struct Reference

Comparison function object for the multimap.

Public Member Functions

- bool [operator\(\)](#) (const pair< double, double > &k1, const pair< double, double > &k2)
- bool [operator\(\)](#) (const pair< double, double > &k1, const pair< double, double > &k2)

4.2.1 Detailed Description

Comparison function object for the multimap.

Compares the Isc of both groups. In case they are equal, compares the SVbr of the groups.

Returns

True in case the Isc of the first element is lower than the first one. If they are equal, returns TRUE if the first element has higher SVbr.

4.2.2 Member Function Documentation

4.2.2.1 operator>() [1/2]

```
bool Classcomp::operator() (
    const pair< double, double > & k1,
    const pair< double, double > & k2 ) [inline]
```

4.2.2.2 operator>() [2/2]

```
bool Classcomp::operator() (
    const pair< double, double > & k1,
    const pair< double, double > & k2 ) [inline]
```

4.3 Gnuplot Class Reference

```
#include <gnuplot.h>
```

Public Member Functions

- [Gnuplot](#) ()
- [~Gnuplot](#) ()
- void [operator](#)() (const string &command)
- [Gnuplot](#) ()
- [~Gnuplot](#) ()
- void [operator](#)() (const string &command)

Protected Attributes

- FILE * [gnuplotpipe](#)

4.3.1 Constructor & Destructor Documentation

4.3.1.1 Gnuplot() [1/2]

```
Gnuplot::Gnuplot ( )
```

4.3.1.2 ~Gnuplot() [1/2]

```
Gnuplot::~~Gnuplot ( )
```

4.3.1.3 Gnuplot() [2/2]

```
Gnuplot::Gnuplot ( )
```

4.3.1.4 ~Gnuplot() [2/2]

```
Gnuplot::~~Gnuplot ( )
```

4.3.2 Member Function Documentation

4.3.2.1 operator() [1/2]

```
void Gnuplot::operator() (
    const string & command )
```

4.3.2.2 operator() [2/2]

```
void Gnuplot::operator() (
    const string & command )
```

4.3.3 Member Data Documentation

4.3.3.1 gnuplotpipe

```
FILE * Gnuplot::gnuplotpipe [protected]
```

4.4 grupString Struct Reference

Structure to gather global information of a group of cells that share, at least, the same short-circuit current.

Public Attributes

- double [Isc](#)
Shortcut current of all the cells in the group.
- double [SVbr](#)
Breakdown voltage of the group.
- double [SVoc](#)
Sum of all the open circuit voltage of the active cells.
- double [SVocr](#)
Sum of all the open circuit voltage of the non-active cells.
- int [N](#)
Number of cells in this group.
- double [Limit](#)
Voltage between the terminals of the panel where either a change in the distribution of the tensions or currents in the panel will take place.

4.4.1 Detailed Description

Structure to gather global information of a group of cells that share, at least, the same short-circuit current.

4.4.2 Member Data Documentation

4.4.2.1 Isc

```
double grupString::Isc
```

Shortcut current of all the cells in the group.

4.4.2.2 Limit

```
double grupString::Limit
```

Voltage between the terminals of the panel where either a change in the distribution of the tensions or currents in the panel will take place.

4.4.2.3 N

```
int grupString::N
```

Number of cells in this group.

4.4.2.4 SVbr

```
double grupString::SVbr
```

Breakdown voltage of the group.

Calculated by adding all the breakdown voltages calculated in the group Vbrx of every cell.

See also

Definition of Vbrx in the Theoretical documentation.

4.4.2.5 SVoc

```
double grupString::SVoc
```

Sum of all the open circuit voltage of the active cells.

Sum of all the open circuit voltage of the cells in the group that are actually driving its short-circuit current (active cells) [V].

4.4.2.6 SVocr

```
double grupString::SVocr
```

Sum of all the open circuit voltage of the non-active cells.

Sum of all the open circuit voltage of the cells in the group that are not driving its short-circuit current (non-active cells), but an inferior current [V].

4.5 inMapCell Struct Reference

Structure with info of groups of cells with the same Isc and Vbrx.

Public Attributes

- [grupString inMapSum](#)

Global information of the group of cells with the same Isc and Vbrx.

- list< pair< int, [grupString](#) > > [inMapDetails](#)

Detailed information of the group of cells with the same Isc and Vbrx.

4.5.1 Detailed Description

Structure with info of groups of cells with the same Isc and Vbrx.

Contains global information (inMapSum) and detailed information (inMapDetails) about groups of cells with the same short-circuit current and breakdown voltage.

See also

[inVectorCell](#)

4.5.2 Member Data Documentation

4.5.2.1 inMapDetails

```
list< pair< int, grupString > > inMapCell::inMapDetails
```

Detailed information of the group of cells with the same Isc and Vbrx.

The cells in the group are split in smaller groups that share the same string. Every entry in the list contains a pair with an integer corresponding to the index of the string and a [grupString](#) structure with the grouped info of this group.

4.5.2.2 inMapSum

```
grupString inMapCell::inMapSum
```

Global information of the group of cells with the same Isc and Vbrx.

Contains the short-circuit current of the group and the total sum of certain parameters. The [grupString](#)'s Limit attribute stored in this structure refers to the internal limits.

These "internal" limits are the total voltage in the panel needed to get every bypass diode in conducting state. In case there's no diode, the limit will match the lower external limit. The internal limits represent a change in the distribution of the total voltage.

See also

[grupString](#)

4.6 inVectorCell Struct Reference

Vector meant to contain info of groups of cells with the same short-circuit current (Isc).

Public Attributes

- [grupString](#) inVectorSum
Global information of the group of cells with the same short-circuit current.
- map< double, [inMapCell](#) > [inVectorDetails](#)
Detailed information of the group of cells with the same short-circuit current.

4.6.1 Detailed Description

Vector meant to contain info of groups of cells with the same short-circuit current (Isc).

Contains global information (inVectorSum) and detailed information (inVectorDetail). Global information refers to the sum of certain parameters. Detailed information distinguish smaller groups that share the same Isc and Vbrx.

See also

[inMapCell](#)

4.6.2 Member Data Documentation

4.6.2.1 inVectorDetails

```
map< double, inMapCell > inVectorCell::inVectorDetails
```

Detailed information of the group of cells with the same short-circuit current.

The cells in the group are split in smaller groups that share the same breakdown voltage calculated in the group Vbrx. Every entry in the map is composed by key, which is a double corresponding to Vbrx, and a [inMapCell](#) structure with the information of this reduced group.

4.6.2.2 inVectorSum

```
grupString inVectorCell::inVectorSum
```

Global information of the group of cells with the same short-circuit current.

Contains the short-circuit current of the group and the total sum of certain parameters. The [grupString](#)'s Limit attribute stored in this structure refers to the external limits. These "external" limits are the total voltage in the panel needed to get every cell into breakdown. The external limits represent a change in the total current.

See also

[grupString](#)

4.7 solar_cell Class Reference

Represents a PV cell, the most basic element of a solar generator.

```
#include <solar_cell.h>
```

Public Member Functions

- [solar_cell](#) (void)
- [solar_cell](#) (const [solar_cell](#) &)
- void [setIndex](#) (int)
- void [setG](#) (double)
- void [setTc](#) (double)
- void [setIo](#) (void)
- void [setIsc](#) (void)
- void [setIph](#) (void)
- void [setVoc](#) (void)
- void [setIcell](#) (double)
- void [setVcell](#) (double)
- void [setVbreak](#) (double)
- int [getIndex](#) (void)
- double [getIo](#) (void)
- double [getIph](#) (void)
- double [getIsc](#) (void)
- double [getVoc](#) (void)
- double [getG](#) (void)
- double [getTc](#) (void)
- double [getIcell](#) (void)
- double [getVcell](#) (void)
- double [getVbreak](#) (void)
- double [calcfcn](#) (void)
- double [calcderi](#) (void)
- double [calcderv](#) (void)
- [solar_cell](#) (void)
Constructor of the class [solar_cell](#).
- [solar_cell](#) (const [solar_cell](#) &)
Constructor of the class [solar_cell](#).
- void [setIndex](#) (int)
Set an integer value for the index.
- void [setG](#) (double)
Set a double value for the irradiance [W/m2].
- void [setTc](#) (double)
Set a double value for the temperature of the cell [°C].
- void [setIo](#) (void)
Updates the value for the reverse saturation current [A] according to the current value of the temperature of the cell Tc.
- void [setIsc](#) (void)
Updates the value for the short-circuit current [A] according to the current values of the temperature of the cell Tc and the irradiance G.
- void [setIph](#) (void)
Updates the value for the photogenerated current [A] according to the current values of the temperature of the cell Tc and the irradiance G.

- void [setVoc](#) (void)
Updates the value for the open circuit voltage [V] according to the current values of the temperature of the cell Tc and the irradiance G.
- void [setIcell](#) (double)
Set a double value for the current [A].
- void [setVcell](#) (double)
Set a double value for the voltage [V].
- void [setVbreak](#) (double)
Set a double value for the breakdown voltage [V].
- int [getIndex](#) (void)
Gets the cell's index.
- double [getIo](#) (void)
Gets the reverse saturation current [A].
- double [getIph](#) (void)
Gets the photogenerated current [A].
- double [getIsc](#) (void)
Gets the short-circuit current [A].
- double [getVoc](#) (void)
Gets the open circuit voltage [V].
- double [getG](#) (void)
Gets the irradiance [W/m2].
- double [getTc](#) (void)
Gets the temperature of the cell [°C].
- double [getIcell](#) (void)
Gets the cell's current [A].
- double [getVcell](#) (void)
Gets the cell's voltage [V].
- double [getVbreak](#) (void)
Gets the breakdown voltage [V].
- double [calcfcn](#) (void)
Calculates the fc function described in the math part of the mainPage.
- double [calcderi](#) (void)
Calculates the partial derivative respect the current of the cell, Icell, of the fc function described in the math part of the mainPage.
- double [calcderv](#) (void)
Calculates the partial derivative respect the voltage of the cell, Vcell, of the fc function described in the math part of the mainPage.

Protected Attributes

- int [index](#)
Index of the cell. Serves as an identifier (ID) of the cell once it is grouped inside a string.
- double [Iph](#)
Photogenerated current [A].
- double [Io](#)
Reverse saturation current [A].
- double [Isc](#)
Short-circuit current [A].
- double [Voc](#)
Open circuit voltage [V].
- double [Tc](#)

- Temperature of the cell [°C].*
- double **G**
- Irradiance [W/m2].*
- double **Icell**
- Current through the cell [A].*
- double **Vcell**
- Voltage between the terminals of the cell [V].*
- double **Vbreak**
- Breakdown voltage [V].*

4.7.1 Detailed Description

Represents a PV cell, the most basic element of a solar generator.

This class contains all the parameters that define a single PV cell and to perform the calculations needed. Only the operational parameters of a PV cell are considered as attributes of this class. For intrinsic parameters of a PV cell, such as those that depend on the PV cell's material, they are implemented as constant and can not be edited. These values correspond to a silicon, multicrystalline PV cell.

The editable attributes of this class are:

- **index** Identifier of the cell.
- **Iph** Photogenerated current [A].
- **Io** Reverse saturation current [A].
- **Isc** Short-circuit current [A].
- **Voc** Open circuit voltage [V].
- **Tc** Temperature of the cell [°C].
- **G** Irradiance [W/m2].
- **Icell** Current [A].
- **Vcell** Voltage [V].
- **Vbreak** Breakdown voltage [V].

When it is not specified in the constructor of the class some attributes are initialized with reference values. These reference values are:

- **Ioref** The reverse saturation current of reference is 1.26E-9 A.
- **Iphref** The photogenerated current of reference is 3.798 A.
- **Iscref** The short-circuit current of reference is 3.798 A.
- **Vocref** The open circuit voltage of reference is 0.9 V.
- **Tcref** The temperature of the cell of reference is 25.0 °C.
- **Gref** The irradiance of reference is 1000 W/m2.

The mathematical models of this library use some constants or approximations. The values used related to this class are:

- α Breakdown alpha parameter: 0.002
- **Vbr** Breakdown voltage: -15.0 V
- **k** Boltzmann constant: 1.38e-23 J/°K
- **SF** Soiling Factor: 1
- **Rs** Total resistance of the cell in series: 0.00895
- **Rsh** Total shunt resistance of the cell: 30.0
- **q** Charge of an electron: 1.602e-19 C
- **a** Temperature coefficient: 0.0004 A/°C
- **B** Voltage temperature coefficient: -0.0023 V/°C
- **m** Breakdown exponent: 3

See also

[solar_string\(\)](#)

Note

The theoretical concepts behind this class are explained in the solarCell_ch section of the mainPage.

Warning

This library contemplates the calculations of solar panels under mismatched conditions where irradiance (G) and temperature of the cell (Tc) are different across the facility. Scenarios where the cells that compose the panels have different intern parameters are NOT in the scope of this library and will not compute.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 solar_cell() [1/4]

```
solar_cell::solar_cell (  
    void )
```

4.7.2.2 solar_cell() [2/4]

```
solar_cell::solar_cell (  
    const solar\_cell & cell )
```

4.7.2.3 solar_cell() [3/4]

```
solar_cell::solar_cell (
    void )
```

Constructor of the class [solar_cell](#).

Uses all the reference values for the attributes.

4.7.2.4 solar_cell() [4/4]

```
solar_cell::solar_cell (
    const solar\_cell & )
```

Constructor of the class [solar_cell](#).

Uses the same attributes as the [solar_cell](#) object introduced as a parameter.

Parameters

<i>&cell</i>	solar_cell object to copy the attributes from.
------------------	----------------------------------------------------------------

4.7.3 Member Function Documentation**4.7.3.1 calcderi()** [1/2]

```
double solar_cell::calcderi (
    void )
```

4.7.3.2 calcderi() [2/2]

```
double solar_cell::calcderi (
    void )
```

Calculates the partial derivative respect the current of the cell, *lcell*, of the *fc* function described in the math part of the mainPage.

It is used to build the jacobian matrix explained in the *math_newton_part2* section. The current values of *Vcell* and *lcell* are used to calculate this function.

Returns

A double type with the value of the partial derivative respect the current of the cell of the funtion *fc*.

See also

[math](#)

4.7.3.3 calcderv() [1/2]

```
double solar_cell::calcderv (
    void )
```

4.7.3.4 calcderv() [2/2]

```
double solar_cell::calcderv (
    void )
```

Calculates the partial derivative respect the voltage of the cell, Vcell, of the fc function described in the math part of the mainPage.

It is used to build the jacobian matrix explained in the math_newton_part2 section. The current values of Vcell and lcell are used to calculate this function.

Returns

A double type with the value of the partial derivative respect the voltage of the cell of the funtion fc.

See also

math

4.7.3.5 calcfcn() [1/2]

```
double solar_cell::calcfcn (
    void )
```

4.7.3.6 calcfcn() [2/2]

```
double solar_cell::calcfcn (
    void )
```

Calculates the fc function described in the math part of the mainPage.

The current values of Vcell and lcell are used to calculate this function.

Returns

A double type with the value of the funtion fc.

See also

math

4.7.3.7 getG() [1/2]

```
double solar_cell::getG (  
    void )
```

4.7.3.8 getG() [2/2]

```
double solar_cell::getG (  
    void )
```

Gets the irradiance [W/m2].

Returns

A double type with the value of the Irradiance [W/m2].

4.7.3.9 getIcell() [1/2]

```
double solar_cell::getIcell (  
    void )
```

4.7.3.10 getIcell() [2/2]

```
double solar_cell::getIcell (  
    void )
```

Gets the cell's current [A].

Returns

A double type with the value of the current [A].

4.7.3.11 getIndex() [1/2]

```
int solar_cell::getIndex (  
    void )
```

4.7.3.12 getIndex() [2/2]

```
int solar_cell::getIndex (
    void )
```

Gets the cell's index.

Returns

An integer type with the value of the index.

4.7.3.13 getIo() [1/2]

```
double solar_cell::getIo (
    void )
```

4.7.3.14 getIo() [2/2]

```
double solar_cell::getIo (
    void )
```

Gets the reverse saturation current [A].

Returns

A double type with the value of the reverse saturation current [A].

4.7.3.15 getIph() [1/2]

```
double solar_cell::getIph (
    void )
```

4.7.3.16 getIph() [2/2]

```
double solar_cell::getIph (
    void )
```

Gets the photogenerated current [A].

Returns

A double type with the value of the photogenerated current [A].

4.7.3.17 getIsc() [1/2]

```
double solar_cell::getIsc (  
    void )
```

4.7.3.18 getIsc() [2/2]

```
double solar_cell::getIsc (  
    void )
```

Gets the short-circuit current [A].

Returns

A double type with the value of the short-circuit current [A].

4.7.3.19 getTc() [1/2]

```
double solar_cell::getTc (  
    void )
```

4.7.3.20 getTc() [2/2]

```
double solar_cell::getTc (  
    void )
```

Gets the temperature of the cell [°C].

Returns

A double type with the value of the temperature of the cell [°C].

4.7.3.21 getVbreak() [1/2]

```
double solar_cell::getVbreak (  
    void )
```

4.7.3.22 getVbreak() [2/2]

```
double solar_cell::getVbreak (
    void )
```

Gets the breakdown voltage [V].

Returns

A double type with the value of the breakdown voltage [V].

4.7.3.23 getVcell() [1/2]

```
double solar_cell::getVcell (
    void )
```

4.7.3.24 getVcell() [2/2]

```
double solar_cell::getVcell (
    void )
```

Gets the cell's voltage [V].

Returns

A double type with the value of the voltage [V].

4.7.3.25 getVoc() [1/2]

```
double solar_cell::getVoc (
    void )
```

4.7.3.26 getVoc() [2/2]

```
double solar_cell::getVoc (
    void )
```

Gets the open circuit voltage [V].

Returns

A double type with the value of the open circuit voltage [V].

4.7.3.27 setG() [1/2]

```
void solar_cell::setG (
    double _G )
```

4.7.3.28 setG() [2/2]

```
void solar_cell::setG (
    double )
```

Set a double value for the irradiance [W/m2].

Parameters

↩	Double value of the new irradiance [W/m2].
↩	
G	

4.7.3.29 setIcell() [1/2]

```
void solar_cell::setIcell (
    double _Icell )
```

4.7.3.30 setIcell() [2/2]

```
void solar_cell::setIcell (
    double )
```

Set a double value for the current [A].

Parameters

_Icell	Double value of the cell's current [A].
---------------	-----------------------------------------

4.7.3.31 setIndex() [1/2]

```
void solar_cell::setIndex (
    int _index )
```

4.7.3.32 setIndex() [2/2]

```
void solar_cell::setIndex (
    int )
```

Set an integer value for the index.

Parameters

<code>_index</code>	Integer number of the index.
---------------------	------------------------------

4.7.3.33 setIo() [1/2]

```
void solar_cell::setIo (
    void )
```

4.7.3.34 setIo() [2/2]

```
void solar_cell::setIo (
    void )
```

Updates the value for the reverse saturation current [A] according to the current value of the temperature of the cell T_c .

4.7.3.35 setIph() [1/2]

```
void solar_cell::setIph (
    void )
```

4.7.3.36 setIph() [2/2]

```
void solar_cell::setIph (
    void )
```

Updates the value for the photogenerated current [A] according to the current values of the temperature of the cell T_c and the irradiance G .

4.7.3.37 setIsc() [1/2]

```
void solar_cell::setIsc (
    void )
```

4.7.3.38 setIsc() [2/2]

```
void solar_cell::setIsc (
    void )
```

Updates the value for the short-circuit current [A] according to the current values of the temperature of the cell T_c and the irradiance G .

4.7.3.39 setTc() [1/2]

```
void solar_cell::setTc (
    double _Tc )
```

4.7.3.40 setTc() [2/2]

```
void solar_cell::setTc (
    double )
```

Set a double value for the temperature of the cell [°C].

Parameters

<code>_Tc</code>	Double value of the new temperature [°C].
------------------	-------------------------------------------

4.7.3.41 setVbreak() [1/2]

```
void solar_cell::setVbreak (
    double _Vbreak )
```

4.7.3.42 setVbreak() [2/2]

```
void solar_cell::setVbreak (
    double )
```

Set a double value for the breakdown voltage [V].

Parameters

<code>_Vbreak</code>	Double value of the cell's breakdown voltage [V].
----------------------	---------------------------------------------------

4.7.3.43 setVcell() [1/2]

```
void solar_cell::setVcell (  
    double _Vcell )
```

4.7.3.44 setVcell() [2/2]

```
void solar_cell::setVcell (  
    double )
```

Set a double value for the voltage [V].

Parameters

<code>_Vcell</code>	Double value of the cell's voltage [V].
---------------------	-----------------------------------------

4.7.3.45 setVoc() [1/2]

```
void solar_cell::setVoc (  
    void )
```

4.7.3.46 setVoc() [2/2]

```
void solar_cell::setVoc (  
    void )
```

Updates the value for the open circuit voltage [V] according to the current values of the temperature of the cell T_c and the irradiance G .

4.7.4 Member Data Documentation

4.7.4.1 G

```
double solar_cell::G [protected]
```

Irradiance [W/m²].

4.7.4.2 Icell

```
double solar_cell::Icell [protected]
```

Current through the cell [A].

4.7.4.3 index

```
int solar_cell::index [protected]
```

Index of the cell. Serves as an identifier (ID) of the cell once it is grouped inside a string.

4.7.4.4 Io

```
double solar_cell::Io [protected]
```

Reverse saturation current [A].

4.7.4.5 Iph

```
double solar_cell::Iph [protected]
```

Photogenerated current [A].

4.7.4.6 Isc

```
double solar_cell::Isc [protected]
```

Short-circuit current [A].

4.7.4.7 Tc

```
double solar_cell::Tc [protected]
```

Temperature of the cell [°C].

4.7.4.8 Vbreak

```
double solar_cell::Vbreak [protected]
```

Breakdown voltage [V].

4.7.4.9 Vcell

```
double solar_cell::Vcell [protected]
```

Voltage between the terminals of the cell [V].

4.7.4.10 Voc

```
double solar_cell::Voc [protected]
```

Open circuit voltage [V].

4.8 solar_string Class Reference

Represents a string of solar cells group under the same bypass diode.

```
#include <solar_string.h>
```

Public Member Functions

- [solar_string](#) (void)
- [solar_string](#) & [operator=](#) (const [solar_string](#) &)
- [~solar_string](#) (void)
- int [getambDiode](#) (void)
- double [getIscmin](#) (void)
- double [getSvoc](#) (void)
- double [getSvbr](#) (void)
- double [getVstring](#) (void)
- double [getSvcell](#) (void)
- double [getVdiode](#) (void)
- void [update_physical](#) (int, const char *)
- void [update_electrical](#) (void)
- void [inici_corda](#) ([solar_cell](#))
- void [setSvoc](#) (void)
- void [setSvbr](#) (void)
- void [setVstring](#) (double)
- void [setSvcell](#) (void)
- void [setIdiode](#) (double)
- double [getIdiode](#) (void)
- void [sort_string](#) (void)
- int [grups](#) (void)
- void [classifica](#) (int, int)
- void [setSVbrx](#) (void)
- void [genLlist](#) (void)
- void [AccLlist](#) (void)
- void [setBounds](#) (void)
- void [findValues](#) (double &, double &)
- double [minim](#) (double *)
- [solar_string](#) (void)

Constructor of the class [solar_string](#).

- [solar_string](#) & [operator=](#) (const [solar_string](#) &)
- [~solar_string](#) (void)

Destructor of the class [solar_string](#).

- int [getambDiode](#) (void)
Indicates whether the string of PV cells has a by-pass diode or not.
- double [getIscmin](#) (void)
Gets the minimum short-circuit current in the string [A].
- double [getSvoc](#) (void)
Gets the sum of all the open circuit voltage of the cells in the string [V].
- double [getSvbr](#) (void)
Gets the sum of the breakdown voltage of all the cells in the string [V].
- double [getVstring](#) (void)
Gets the voltage between the terminals of the string [V].
- double [getSvcell](#) (void)
Gets the sum of the voltage between the terminals of every cell in the string [V].
- double [getVdiode](#) (void)
Gets the voltage between the terminals of the bypass diode [V].
- void [update_physical](#) (int, const char *)
Reads a specified document and uses its information to update the fields of Irradiance and Temperature of the specified string.
- void [update_electrical](#) (void)

Sets an index for every cells and updated their electrical parameters according to the current values of temperature and irradiance.

- void [inici_corda](#) ([solar_cell](#))

Set all the cells in the string are like the one provided as parameter.

- void [setSvoc](#) (void)

Updates the Svoc attribute with the current value of Voc of every cell.

- void [setSvbr](#) (void)

Updates the Svbreak attribute with the current value of Vbreak of every cell.

- void [setVstring](#) (double)

Set a new value for the voltage between the terminals of the string.

- void [setSvcell](#) (void)

Updates the value of the sum of the voltage between the terminals of every cell in the string (Svcell) with its current value.

- void [setIdiode](#) (double)

- double [getIdiode](#) (void)

- void [sort_string](#) (void)

- int [grups](#) (void)

- void [classifica](#) (int, int)

- void [setSVbrx](#) (void)

Updates the value of the sum of the breakdown voltage of all the cells in the string.

- void [genList](#) (void)

Fill the CellsGr list with the different groups of cells under the same working conditions.

- void [AccList](#) (void)

Once the CellsGr has been filled with all the cells in the string, and it has been sorted, this method groups them under the same working conditions.

- void [setBounds](#) (void)

- void [findValues](#) (double &, double &)

Approximates the initial values for the iterative method.

- double [minim](#) (double *)

Looks for the minimum of an array of double type pointers.

Public Attributes

- int [midaCorda](#)

Number of cells contained in the string.

- double [Vdiode](#)

Voltage between bypass diode terminals [V].

- double [Idiode](#)

Current through the bypass diode [A].

- [solar_cell](#) * [corda](#)

Array of [solar_cell](#) objects.

- [bypass_diode](#) [rm3_c](#)

[bypass_diode](#) object.

- list< [TableStr](#) > [CellsGr](#)

List that contains all the info about the different groups of cells in the string that share the same short-circuit current *Isc*.

4.8.1 Detailed Description

Represents a string of solar cells group under the same bypass diode.

However, the diode can be missing, broken or non-active. This class contains all the details of the components of the string. The cells in the string are divided in groups according to their short-circuit current I_{sc} . These groups are distinguished:

- **Active cells groups:** Cells working under their own short-circuit current I_{sc} .
- **Non-active cells groups:** Cells working under a different current from their I_{sc} (a lower value).
- **Breakdown cells groups:** Cells working in the breakdown zone of the cell. Therefore working under a different current from their I_{sc} (a higher value).

Given the total current and voltage between the terminals of the string, this class can find an initial estimation of the state of every component. The values used are the following:

- **Non-active cells:** Current is imposed by the rest of the panel or an active group in the string. Working voltage is its open circuit voltage.
- **Breakdown cells:** Current is imposed by the rest of the panel or an active group in the string. Working voltage is its breakdown voltage.
- **Active cells:** Current is its short-circuit current. Voltage is deducted from the total voltage in the string, the diode's voltage and voltage in the rest of the groups.

See also

[solar_cell](#)

[bypass_diode](#)

Note

The theoretical concepts behind this class are explained in the `string_ch` section of the `mainPage`.

4.8.2 Constructor & Destructor Documentation

4.8.2.1 `solar_string()` [1/2]

```
solar_string::solar_string (  
    void )
```

4.8.2.2 `~solar_string()` [1/2]

```
solar_string::~solar_string (  
    void )
```

4.8.2.3 solar_string() [2/2]

```
solar_string::solar_string (  
    void )
```

Constructor of the class [solar_string](#).

Uses all the reference values for the attributes.

4.8.2.4 ~solar_string() [2/2]

```
solar_string::~~solar_string (  
    void )
```

Destructor of the class [solar_string](#).

4.8.3 Member Function Documentation

4.8.3.1 AccLlist() [1/2]

```
void solar_string::AccLlist (  
    void )
```

4.8.3.2 AccLlist() [2/2]

```
void solar_string::AccLlist (  
    void )
```

Once the CellsGr has been filled with all the cells in the string, and it has been sorted, this method groups them under the same working conditions.

This method is used inside [genLlist\(\)](#).

4.8.3.3 classifica() [1/2]

```
void solar_string::classifica (  
    int ,  
    int )
```

4.8.3.4 classifica() [2/2]

```
void solar_string::classifica (
    int ,
    int )
```

Warning

NOT IMPLEMENTED TODO: delete

4.8.3.5 findValues() [1/2]

```
void solar_string::findValues (
    double & Iin,
    double & Vin )
```

4.8.3.6 findValues() [2/2]

```
void solar_string::findValues (
    double & ,
    double & )
```

Approximates the initial values for the iterative method.

Given the total current and voltage between terminals in the string, assigns the electrical working point of every component in the string. This function first finds the state of the bypass diode and then the state and electrical conditions of every component. No value is returned, the changes are done in the [solar_cell](#) and [bypass_diode](#) objects contained by the string object.

Parameters

<i>&lin</i>	Total current through the string [A].
<i>&Vin</i>	Total potential difference between terminals of the string [V].

4.8.3.7 genLlist() [1/2]

```
void solar_string::genLlist (
    void )
```


4.8.3.8 genLlist() [2/2]

```
void solar_string::genLlist (
    void )
```

Fill the CellsGr list with the different groups of cells under the same working conditions.

It also calculates the corresponding electrical parameters of each group.

4.8.3.9 getambDiode() [1/2]

```
int solar_string::getambDiode (
    void )
```

4.8.3.10 getambDiode() [2/2]

```
int solar_string::getambDiode (
    void )
```

Indicates whether the string of PV cells has a by-pass diode or not.

By default it is 1.

Returns

An integer data type. 1 indicates that there is a diode, 0 indicates that there is not.

4.8.3.11 getIdiode() [1/2]

```
double solar_string::getIdiode (
    void )
```

4.8.3.12 getIdiode() [2/2]

```
double solar_string::getIdiode (
    void )
```

Warning

NOT IMPLEMENTED TODO: delete

4.8.3.13 getIscmin() [1/2]

```
double solar_string::getIscmin (  
    void )
```

4.8.3.14 getIscmin() [2/2]

```
double solar_string::getIscmin (  
    void )
```

Gets the minimum short-circuit current in the string [A].

Returns

Double data type with the value of the minimum short-circuit current [A].

4.8.3.15 getSvbr() [1/2]

```
double solar_string::getSvbr (  
    void )
```

4.8.3.16 getSvbr() [2/2]

```
double solar_string::getSvbr (  
    void )
```

Gets the sum of the breakdown voltage of all the cells in the string [V].

Returns

Double data type with the value of the sum of breakdown voltages [V].

4.8.3.17 getSvcell() [1/2]

```
double solar_string::getSvcell (  
    void )
```

4.8.3.18 getSvcell() [2/2]

```
double solar_string::getSvcell (  
    void )
```

Gets the sum of the voltage between the terminals of every cell in the string [V].

This value can be different than the obtained with getVstring.

Returns

Double data type with the value of the sum of the voltages in every cell [V].

4.8.3.19 getSvoc() [1/2]

```
double solar_string::getSvoc (  
    void )
```

4.8.3.20 getSvoc() [2/2]

```
double solar_string::getSvoc (  
    void )
```

Gets the sum of all the open circuit voltage of the cells in the string [V].

Returns

Double data type with the value of the sum of open circuit voltages [V].

4.8.3.21 getVdiode() [1/2]

```
double solar_string::getVdiode (  
    void )
```

4.8.3.22 getVdiode() [2/2]

```
double solar_string::getVdiode (  
    void )
```

Gets the voltage between the terminals of the bypass diode [V].

Returns

Double data type value of the voltage in the bypass diode [V].

4.8.3.23 getVstring() [1/2]

```
double solar_string::getVstring (
    void )
```

4.8.3.24 getVstring() [2/2]

```
double solar_string::getVstring (
    void )
```

Gets the voltage between the terminals of the string [V].

Returns

Double data type with the value of the voltage in the string [V].

4.8.3.25 grups() [1/2]

```
int solar_string::grups (
    void )
```

4.8.3.26 grups() [2/2]

```
int solar_string::grups (
    void )
```

Warning

NOT IMPLEMENTED TODO: delete

4.8.3.27 inici_corda() [1/2]

```
void solar_string::inici_corda (
    solar_cell sc )
```

4.8.3.28 inici_corda() [2/2]

```
void solar_string::inici_corda (
    solar_cell )
```

Set all the cells in the string are like the one provided as parameter.

Fills the array *corda with as many [solar_cell](#) objects as indicated in midaCorda.

Parameters

sc	solar_cell object that represents all the cells in the string.
----	--------------------------------------------------------------------------------

See also

[solarCell_ch](#)

4.8.3.29 minim() [1/2]

```
double solar_string::minim (
    double * Fa )
```

4.8.3.30 minim() [2/2]

```
double solar_string::minim (
    double * )
```

Looks for the minimum of an array of double type pointers.

Parameters

*Fa	Array of double pointers.
-----	---------------------------

Returns

The minimum of the elements in the array.

4.8.3.31 operator=() [1/2]

```
solar_string& solar_string::operator= (
    const solar_string & )
```

4.8.3.32 operator=() [2/2]

```
solar_string& solar_string::operator= (
    const solar_string & )
```

4.8.3.33 setBounds() [1/2]

```
void solar_string::setBounds (  
    void )
```

4.8.3.34 setBounds() [2/2]

```
void solar_string::setBounds (  
    void )
```

Warning

NOT IMPLEMENTED TODO: delete

4.8.3.35 setIdiode() [1/2]

```
void solar_string::setIdiode (  
    double )
```

4.8.3.36 setIdiode() [2/2]

```
void solar_string::setIdiode (  
    double )
```

Warning

NOT IMPLEMENTED TODO: delete

4.8.3.37 setSvbr() [1/2]

```
void solar_string::setSvbr (  
    void )
```

4.8.3.38 setSvbr() [2/2]

```
void solar_string::setSvbr (  
    void )
```

Updates the Svbreak attribute with the current value of Vbreak of every cell.

4.8.3.39 setSVbrx() [1/2]

```
void solar_string::setSVbrx (
    void )
```

4.8.3.40 setSVbrx() [2/2]

```
void solar_string::setSVbrx (
    void )
```

Updates the value of the sum of the breakdown voltage of all the cells in the string.

If there is no bypass diode in the string, it is equal to the sum of the voltage between the terminals of every cell. If there is a bypass diode, it is obtained as explained in the theoretical documentation.

4.8.3.41 setSvcell() [1/2]

```
void solar_string::setSvcell (
    void )
```

4.8.3.42 setSvcell() [2/2]

```
void solar_string::setSvcell (
    void )
```

Updates the value of the sum of the voltage between the terminals of every cell in the string (Svcell) with its current value.

It does the sum again. In case any value of any cell has changed.

4.8.3.43 setSvoc() [1/2]

```
void solar_string::setSvoc (
    void )
```

4.8.3.44 setSvoc() [2/2]

```
void solar_string::setSvoc (
    void )
```

Updates the Svoc attribute with the current value of Voc of every cell.

4.8.3.45 setVstring() [1/2]

```
void solar_string::setVstring (
    double _Vstring )
```

4.8.3.46 setVstring() [2/2]

```
void solar_string::setVstring (
    double )
```

Set a new value for the voltage between the terminals of the string.

Parameters

<i>Vstring</i>	New voltage between the terminals of the string [V].
----------------	------------------------------------------------------

4.8.3.47 sort_string() [1/2]

```
void solar_string::sort_string (
    void )
```

4.8.3.48 sort_string() [2/2]

```
void solar_string::sort_string (
    void )
```

Warning

NOT IMPLEMENTED TODO: delete

4.8.3.49 update_electrical() [1/2]

```
void solar_string::update_electrical (
    void )
```


4.8.3.50 update_electrical() [2/2]

```
void solar_string::update_electrical (
    void )
```

Sets an index for every cells and updated their electrical parameters according to the current values of temperature and irradiance.

4.8.3.51 update_physical() [1/2]

```
void solar_string::update_physical (
    int m,
    const char * filename )
```

4.8.3.52 update_physical() [2/2]

```
void solar_string::update_physical (
    int ,
    const char * )
```

Reads a specified document and uses its information to update the fields of Irradiance and Temperature of the specified string.

Parameters

<i>m</i>	Number (int) of the string to be updated.
<i>*filename</i>	Full path of the file or relative path from the project's directory. TODO: Change to full path only.

Attention

The document must be .csv file with a certain configuration. Please, check the User's Guide to see the guide template for the input document format.

See also

Input file format

4.8.4 Member Data Documentation

4.8.4.1 CellsGr

```
list< TableStr > solar_string::CellsGr
```

List that contains all the info about the different groups of cells in the string that share the same short-circuit current Isc.

Every element in the list is a [TableStr](#) struct with the info of the group of cells.

See also

[TableStr](#) structure

4.8.4.2 corda

```
solar_cell * solar_string::corda
```

Array of [solar_cell](#) objects.

This is a representation of the PV cells contained in this string. The cells in this array must have the same manufacturing properties, but the electrical or physical working values may differ.

See also

[solar_cell](#)

4.8.4.3 Idiode

```
double solar_string::Idiode
```

Current through the bypass diode [A].

4.8.4.4 midaCorda

```
int solar_string::midaCorda
```

Number of cells contained in the string.

4.8.4.5 rm3_c

`bypass_diode` `solar_string::rm3_c`

`bypass_diode` object.

Represents the bypass diode of the string.

4.8.4.6 Vdiode

`double` `solar_string::Vdiode`

Voltage between bypass diode terminals [V].

4.9 solpan Struct Reference

```
#include <solar_panel.h>
```

Public Attributes

- `int` `numStrings`
- `solar_string` `panel` [`numStrings`]
- `double` `Vp`

4.9.1 Member Data Documentation

4.9.1.1 numStrings

`int` `solpan::numStrings`

4.9.1.2 panel

`solar_string` `solpan::panel` [`numStrings`]

4.9.1.3 Vp

`double` `solpan::Vp`

4.10 TableStr Struct Reference

Defines the total parameters of a group of PV cells in the same string that have the same short-circuit current.

```
#include <solar_string.h>
```

Public Attributes

- `vector< int > index`
Vector with the physical position index of every cell in the group.
- `double IscGrup`
Shortcut current of every cell in this group.
- `double SVbr`
Sum of all breakdown voltages of the PV cells in this group.
- `double SVoc`
Sum of all open circuit voltages of the PV cells in this group.
- `double SVbrx`
Sum of all breakdown voltages of the PV cells calculated on the group.

4.10.1 Detailed Description

Defines the total parameters of a group of PV cells in the same string that have the same short-circuit current.

4.10.2 Member Data Documentation

4.10.2.1 [index](#)

```
vector< int > TableStr::index
```

Vector with the physical position index of every cell in the group.

The size of this vector is equal to the number of cells included in this group.

4.10.2.2 [IscGrup](#)

```
double TableStr::IscGrup
```

Shortcut current of every cell in this group.

4.10.2.3 SVbr

```
double TableStr::SVbr
```

Sum of all breakdown voltages of the PV cells in this group.

4.10.2.4 SVbrx

```
double TableStr::SVbrx
```

Sum of all breakdown voltages of the PV cells calculated on the group.

See also

[Solar string's theoretical documentation](#)

4.10.2.5 SVoc

```
double TableStr::SVoc
```

Sum of all open circuit voltages of the PV cells in this group.

Index

- ~Gnuplot
 - Gnuplot, [14](#), [15](#)
- ~solar_string
 - solar_string, [38](#), [39](#)
- AccLlist
 - solar_string, [39](#)
- bypass_diode, [7](#)
 - bypass_diode, [8](#)
 - calcderv, [9](#)
 - calcdiode, [9](#)
 - getIdiode, [10](#)
 - getIr, [10](#)
 - getTc, [11](#)
 - ldiode, [12](#)
 - Ir, [13](#)
 - setIdiode, [11](#)
 - setIr, [12](#)
 - setTc, [12](#)
 - Tc, [13](#)
- calc_ivm
 - Non-member functions, [5](#)
- calcderi
 - solar_cell, [24](#)
- calcderv
 - bypass_diode, [9](#)
 - solar_cell, [24](#), [25](#)
- calcfcn
 - solar_cell, [25](#)
- calcdiode
 - bypass_diode, [9](#)
- CellsGr
 - solar_string, [49](#)
- Classcomp, [13](#)
 - operator(), [13](#), [14](#)
- classifica
 - solar_string, [39](#)
- corda
 - solar_string, [50](#)
- findValues
 - solar_string, [40](#)
- G
 - solar_cell, [33](#)
- genLlist
 - solar_string, [40](#)
- getambDiode
 - solar_string, [41](#)
- getG
 - solar_cell, [25](#), [26](#)
- getCell
 - solar_cell, [26](#)
- getIdiode
 - bypass_diode, [10](#)
 - solar_string, [41](#)
- getIndex
 - solar_cell, [26](#)
- getIo
 - solar_cell, [27](#)
- getIph
 - solar_cell, [27](#)
- getIr
 - bypass_diode, [10](#)
- getIsc
 - solar_cell, [27](#), [28](#)
- getIscmin
 - solar_string, [41](#), [42](#)
- getSvbr
 - solar_string, [42](#)
- getSvcell
 - solar_string, [42](#)
- getSvoc
 - solar_string, [43](#)
- getTc
 - bypass_diode, [11](#)
 - solar_cell, [28](#)
- getVbreak
 - solar_cell, [28](#)
- getVcell
 - solar_cell, [29](#)
- getVdiode
 - solar_string, [43](#)
- getVoc
 - solar_cell, [29](#)
- getVstring
 - solar_string, [43](#), [44](#)
- Gnuplot, [14](#)
 - ~Gnuplot, [14](#), [15](#)
 - Gnuplot, [14](#)
 - gnuplotpipe, [15](#)
 - operator(), [15](#)
- gnuplotpipe
 - Gnuplot, [15](#)
- grups
 - solar_string, [44](#)
- grupString, [15](#)
 - Isc, [16](#)

- Limit, 16
- N, 16
- SVbr, 16
- SVoc, 17
- SVocr, 17
- lcell
 - solar_cell, 34
- ldiode
 - bypass_diode, 12
 - solar_string, 50
- index
 - solar_cell, 34
 - TableStr, 52
- inici_corda
 - solar_string, 44
- inMapCell, 17
 - inMapDetails, 18
 - inMapSum, 18
- inMapDetails
 - inMapCell, 18
- inMapSum
 - inMapCell, 18
- inVectorCell, 18
 - inVectorDetails, 19
 - inVectorSum, 19
- inVectorDetails
 - inVectorCell, 19
- inVectorSum
 - inVectorCell, 19
- lo
 - solar_cell, 34
- lph
 - solar_cell, 34
- lr
 - bypass_diode, 13
- lsc
 - grupString, 16
 - solar_cell, 34
- lscGrup
 - TableStr, 52
- Limit
 - grupString, 16
- midaCorda
 - solar_string, 50
- minim
 - solar_string, 45
- N
 - grupString, 16
- Non-member functions, 5
 - calc_ivm, 5
- numStrings
 - solpan, 51
- operator()
 - Classcomp, 13, 14
- Gnuplot, 15
- operator=
 - solar_string, 45
- panel
 - solpan, 51
- rm3_c
 - solar_string, 50
- setBounds
 - solar_string, 45, 46
- setG
 - solar_cell, 29, 30
- setlcell
 - solar_cell, 30
- setldiode
 - bypass_diode, 11
 - solar_string, 46
- setIndex
 - solar_cell, 30
- setlo
 - solar_cell, 31
- setlph
 - solar_cell, 31
- setlr
 - bypass_diode, 12
- setlsc
 - solar_cell, 31, 32
- setSvbr
 - solar_string, 46
- setSVbrx
 - solar_string, 46, 47
- setSvcell
 - solar_string, 47
- setSvoc
 - solar_string, 47
- setTc
 - bypass_diode, 12
 - solar_cell, 32
- setVbreak
 - solar_cell, 32
- setVcell
 - solar_cell, 33
- setVoc
 - solar_cell, 33
- setVstring
 - solar_string, 47, 48
- solar_cell, 20
 - calcderi, 24
 - calcderv, 24, 25
 - calcfcn, 25
 - G, 33
 - getG, 25, 26
 - getlcell, 26
 - getIndex, 26
 - getlo, 27
 - getlph, 27
 - getlsc, 27, 28

- getTc, [28](#)
- getVbreak, [28](#)
- getVcell, [29](#)
- getVoc, [29](#)
- Icell, [34](#)
- index, [34](#)
- Io, [34](#)
- Iph, [34](#)
- Isc, [34](#)
- setG, [29](#), [30](#)
- setIcell, [30](#)
- setIndex, [30](#)
- setIo, [31](#)
- setIph, [31](#)
- setIsc, [31](#), [32](#)
- setTc, [32](#)
- setVbreak, [32](#)
- setVcell, [33](#)
- setVoc, [33](#)
- solar_cell, [23](#), [24](#)
- Tc, [34](#)
- Vbreak, [35](#)
- Vcell, [35](#)
- Voc, [35](#)
- solar_string, [35](#)
 - ~solar_string, [38](#), [39](#)
- AccLlist, [39](#)
- CellsGr, [49](#)
- classifica, [39](#)
- corda, [50](#)
- findValues, [40](#)
- genLlist, [40](#)
- getambDiode, [41](#)
- getIdiode, [41](#)
- getIscmin, [41](#), [42](#)
- getSvbr, [42](#)
- getSvcell, [42](#)
- getSvoc, [43](#)
- getVdiode, [43](#)
- getVstring, [43](#), [44](#)
- grups, [44](#)
- Idiode, [50](#)
- inici_corda, [44](#)
- midaCorda, [50](#)
- minim, [45](#)
- operator=, [45](#)
- rm3_c, [50](#)
- setBounds, [45](#), [46](#)
- setIdiode, [46](#)
- setSvbr, [46](#)
- setSVbrx, [46](#), [47](#)
- setSvcell, [47](#)
- setSvoc, [47](#)
- setVstring, [47](#), [48](#)
- solar_string, [38](#)
- sort_string, [48](#)
- update_electrical, [48](#)
- update_physical, [49](#)
- Vdiode, [51](#)
- solpan, [51](#)
 - numStrings, [51](#)
 - panel, [51](#)
 - Vp, [51](#)
- sort_string
 - solar_string, [48](#)
- SVbr
 - grupString, [16](#)
 - TableStr, [52](#)
- SVbrx
 - TableStr, [53](#)
- SVoc
 - grupString, [17](#)
 - TableStr, [53](#)
- SVocr
 - grupString, [17](#)
- TableStr, [52](#)
 - index, [52](#)
 - IscGrup, [52](#)
 - SVbr, [52](#)
 - SVbrx, [53](#)
 - SVoc, [53](#)
- Tc
 - bypass_diode, [13](#)
 - solar_cell, [34](#)
- update_electrical
 - solar_string, [48](#)
- update_physical
 - solar_string, [49](#)
- Vbreak
 - solar_cell, [35](#)
- Vcell
 - solar_cell, [35](#)
- Vdiode
 - solar_string, [51](#)
- Voc
 - solar_cell, [35](#)
- Vp
 - solpan, [51](#)