

*This documentation is mostly generated by ChatGPT.*

## Symars Documentation

### 1. Enum DType

- **Description:** Specifies the numeric type (f32 or f64) used for computations in the generated Rust code.

### 2. class GenScalar

- **Description:** Generates Rust functions for scalar SymPy expressions.

#### 2.1. Constructor:

```
def __init__(
    self,
    dtype: DType,
    tol: float = 1e-9,
    precision_digit: int = 20,
    debug: bool = False,
):
```

- dtype (DType): A DType instance specifying the numeric type (f32 or f64).
- tol (float, optional): Tolerance for float comparisons. Default: 1e-9.
- precision\_digit (int, optional): Number of digits to keep when evaluating constants like  $\pi$  or  $\gamma = \lim_{n \rightarrow \infty} (\sum_{k=1}^n \frac{1}{k} - \ln n)$ . Default: 20.
- debug (bool, optional): If True, enables debug output. Default: False.

#### 2.2. Public Methods:

```
generate(func_name: str, expr: sympy.Expr) -> str
```

- func\_name (str): The name of the generated Rust function.
- expr (sympy.Expr): A scalar SymPy expression.
- **Returns:** A string containing the generated Rust function.

```
generate_func_given_params(func_name: str, expr: sympy.Expr, params: List[str]) -> str
```

- `func_name` (str): The name of the generated Rust function.
- `expr` (sympy.Expr): A scalar SymPy expression.
- `params` (List[str]): A list of parameter names for the Rust function.
- **Returns:** A string containing the generated Rust function.

### 3. class GenNalgebra

- **Description:** Generates Rust functions for SymPy matrices using the nalgebra crate.

#### 3.1. Constructor:

- Same as GenScalar.

#### 3.2. Public Methods:

```
generate(func_name: str, mat: sympy.Matrix) -> str
```

- **Description:** Generates a Rust function for the matrix compatible with `nalgebra::SMatrix`.
- `mat` (sympy.Matrix): The SymPy matrix to generate code for.
- `func_name` (str): The name of the generated Rust function.
- **Returns:** A string containing the generated Rust function.

### 4. class GenArrayVec

- **Description:** Generates Rust functions for array-based vector representations.

#### 4.1. Constructor:

- Same as GenScalar.

#### 4.2. Public Methods:

```
generate(func_name: str, mat: sympy.Matrix) -> str
```

- **Description:** Generates Rust code to store the matrix as a flattened vector.
- `mat` (sympy.Matrix): The SymPy matrix to generate code for.
- `func_name` (str): The name of the generated Rust function.

- **Returns:** A string containing the generated Rust code.

## 5. class GenFaer

- **Description:** Generates Rust functions for SymPy matrices using the faer crate.

### 5.1. Constructor:

- Same as GenScalar.

### 5.2. Public Methods:

`generate(func_name: str, mat: sympy.Matrix) -> str`

- **Description:** Generates a Rust function for the matrix compatible with `faer::MatMut`.
- `mat (sympy.Matrix)`: The SymPy matrix to generate code for.
- `func_name (str)`: The name of the generated Rust function.
- **Returns:** A string containing the generated Rust function.

## 6. class GenFaerVec

- **Description:** Generates Rust functions for SymPy vectors using the faer crate.
  - Note: `faer::Col`, `faer::Row`, and `faer::Mat` are distinct types.

### 6.1. Constructor:

- Same as GenScalar.

### 6.2. Public Methods:

- `generate(func_name: str, mat: sympy.Matrix) -> str`
  - **Description:** Generates Rust code for SymPy vector representations.
  - `mat (sympy.Matrix)`: The SymPy matrix or vector to generate code for.
  - `func_name (str)`: The name of the generated Rust function.
  - **Returns:** A string containing the generated Rust code.

## 7. class GenSparse

- **Description:** Generates Rust functions for triplet representations of sparse matrices.

## 7.1. Constructor:

- Same as GenScalar.

## 7.2. Public Methods:

`generate(exprs: list[sympy.Expr], func_name: str) -> str`

- **Description:** Generates Rust functions for sparse representations.
- `mat (sympy.Matrix)`: The SymPy matrix to generate code for.
- `func_name (str)`: The name of the generated Rust function.
- **Returns:** A string containing the generated Rust code.

## 8. class GenDense

- **Description:** Generates Rust functions for dense matrices. **Not user-facing; inspect only for debugging purposes.**

### 8.1. Constructor:

- Same as GenScalar.

### 8.2. Public Methods:

`generate(func_name: str, mat: sympy.Matrix) -> str`

- **Description:** Generates Rust functions to represent the entries of a dense matrix.
- `mat (sympy.Matrix)`: The SymPy matrix to generate code for.
- `func_name (str)`: The name of the generated Rust function.
- **Returns:** A string containing the generated Rust function.

## Appendix: Semantics

### sp.sign

`sp.sign` is implemented to return **itself** with input `+0.0` **and** `-0.0`.

Its semantics is preserved in Symars for the sake of correctness, as some function has sign function in their derivatives. For example, it generates

```
if x.abs() == 0.0_f64 { x } else { x.signum() }
```

rather than `x.signum()`.