**conj() — Complex conjugate**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
<th>Remarks and examples</th>
<th>Conformability</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**Syntax**

```
numeric matrix   conj(numeric matrix Z)
void             _conj(numeric matrix A)
```

---

**Description**

`conj(Z)` returns the elementwise complex conjugate of `Z`, that is, `conj(a+bi) = a - bi`. `conj()` may be used with real or complex matrices. If `Z` is real, `Z` is returned unmodified.

`_conj(A)` replaces `A` with `conj(A)`. Coding `_conj(A)` is equivalent to coding `A = conj(A)`, except that less memory is used.

---

**Remarks and examples**

Given `m × n` matrix `Z`, `conj(Z)` returns an `m × n` matrix; it does not return the transpose. To obtain the conjugate transpose matrix, also known as the adjoint matrix, adjugate matrix, Hermitian adjoin, or Hermitian transpose, code

```
Z'
```

See [M-2] op_transpose.

A matrix equal to its conjugate transpose is called Hermitian or self-adjoint, although in this manual, we often use the term symmetric.

---

**Conformability**

- `conj(Z)`:
  - `Z`: `r × c`
  - `result`: `r × c`

- `_conj(A)`:
  - `input`: `A`: `r × c`
  - `output`: `A`: `r × c`
**Diagnostics**

`conj(Z)` returns a real matrix if `Z` is real and a complex matrix if `Z` is complex.

`conj(Z)`, if `Z` is real, returns `Z` itself and not a copy. This makes `conj()` execute instantly when applied to real matrices.

`conj(A)` does nothing if `A` is real (and hence, does not abort if `A` is a view).

**Also see**

[M-5] `_transpose()` — Transposition in place

[M-4] `scalar` — Scalar mathematical functions