all( ) — Element comparisons

**Description**

all(L) is equivalent to \( \sum(!L) == 0 \) but is significantly faster.

any(L) is equivalent to \( \sum(L) != 0 \) but is slightly faster.

allof(P, s) returns 1 if every element of P equals s and returns 0 otherwise. allof(P, s) is faster and consumes less memory than the equivalent construction \( \text{all}(P == s) \).

anyof(P, s) returns 1 if any element of P equals s and returns 0 otherwise. anyof(P, s) is faster and consumes less memory than the equivalent \( \text{any}(P == s) \).

**Syntax**

```plaintext
real scalar all(real matrix L)
real scalar any(real matrix L)
real scalar allof(transmorphic matrix P, transmorphic scalar s)
real scalar anyof(transmorphic matrix P, transmorphic scalar s)
```

**Remarks and examples**

These functions are fast, so their use is encouraged over alternative constructions.

all( ) and any( ) are typically used with logical expressions to detect special cases, such as

```plaintext
if (any(x < 0)) {
    ...
}
```

or

```plaintext
if (all(x >= 0)) {
    ...
}
```

allof( ) and anyof( ) are used to look for special values:

```plaintext
if (allof(x, 0)) {
    ...
}
```
or

```plaintext
if (anyof(x, 0)) {
  ...
}
```

Do not use `allof()` and `anyof()` to check for missing values—for example, `anyof(x, .)`—because to really check, you would have to check not only . but also .a, .b, ..., .z. Instead use `missing()`; see [M-5] `missing()`.

**Conformability**

`all(L), any(L):`

- `L:      r × c`
- `result: 1 × 1`

`allof(P, s), anyof(P, s):`

- `P:      r × c`
- `s:      1 × 1`
- `result: 1 × 1`

**Diagnostics**

- `all(L)` and `any(L)` treat missing values in `L` as true.
- `all(L)` and `any(L)` return 0 (false) if `L` is `r × 0`, `0 × c`, or `0 × 0`.
- `allof(P, s)` and `anyof(P, s)` return 0 (false) if `P` is `r × 0`, `0 × c`, or `0 × 0`.

**Also see**

[M-4] [Utility — Matrix utility functions](/M-4/Utility)