		X	Y	Z
	separator			
!	.' (transpose) / permute	rot90	system	full
"	for	repmat	repelem (run-length decoding)	blanks
#	specify outputs	display stack (debug)	sound, soundsc, audiowrite	fopen, fwrite, fclose
$	specify inputs		char(vpa(...))	fopen, fread, fclose
%	comment	class	cast	typecast
&	alternative default input/output spec	intersect	and	bitand
'	*Not used. String delimiter*		run-length encoding	now / clock
(	( ) assignment indexing	{ } assignment indexing	( ) assignment ind. with final :	( ) assignment ind. with initial :
)	( ) reference indexing / split	{ } reference indexing	( ) reference ind. with final : / split	( ) refererence ind. with initial : / split
*	.*	kron	matrix product	Cartesian product
+	+		conv2	conv2(…, 'same') / cconv
,	do twice	cos	sin	tan
-	-	setdiff	deconv	
.	break	continue	pause	bitget
/	./	angle	right matrix divide	unwrap
0	*Not used*	predefined literals	predefined literals	
1	*Not used*	predefined literals	predefined literals	
2	*Not used*	predefined literals	predefined literals	
3	*Not used*	predefined literals	predefined literals	
4	*Not used*	predefined literals	predefined literals	
5	*Not used*	predefined literals	predefined literals	
6	*Not used*	predefined literals	predefined literals	
7	*Not used*	predefined literals		
8	*Not used*	predefined literals		
9	*Not used*	predefined literals		
:	colon (range)	linearize array	comma-separated list	bitset
;		acos	asin	atan2
<	<	min	cummin	
=	==	isequal	strcmp	
>	>	max	cummax	
?	if		why	sparse
@	"for" / "do twice" value / "while" index	"for" index	perms	randperm
A	all	all(…, 1)	dec2base. Larger base, any symbols	base2dec. Larger base, any symbols
B	logical(dec2bin(...)-'0')	bin2dec(char(...+'0'))	dec2bin	bin2dec
C			im2col	im2col(…, 'distinct')
D	disp(num2str(…, ...)) / mat2str	disp(num2str(…))	sprintf / fprintf	disp
E	multiply by 2	replace elements in array		
F	*Not used. False (literal)*		exponents of prime factorization	fft, nfft
G	Paste from clipboard G (user-input)	plot	imwrite / imagesc / image / imshow	appearance of graphics / format
H	Paste from clipboard H	Copy to clipboard H		
I	Paste from clipboard I	Copy to clipboard I	col2im	image processing functions
J	Paste from clipboard J	Copy to clipboard J		
K	Paste from clipboard K	Copy to clipboard K		
L	Paste from clipboard L (multi-level)	Copy to clipboard L (multi-level)	gallery	
M	Paste from clipboard M (function-input)	mode		
N	stack size	nchoosek (array)	NaN	isnan
O	zeros	datestr	datenum	datevec
P	flip	flipud	pi	pdist2
Q	increment by 1	accumarray	rat	polyval / roots / polyfit / inpolygon
R	triu	triu(…,1) / build matrix	tril	tril(…,-1) / build matrix
S	sort	sortrows	circshift	sign / fftshift / linspace
T	*Not used. True (literal)*		toeplitz	
U	str2num / string to array / square	str2double		
V	num2str			
W	2 raised to input			
X	*Not used*	regexp	regexprep	
Y	*Not used*		inf	isinf
Z	*Not used*			
[	*Not used. Array delimiter*	ind2sub		
\	mod	mod(…-1)+1	left matrix divide	divisors
]	end (loops or conditional branches)	sub2ind		
^	.^	sqrt	matrix power, or sum of matrix powers	Cartesian power
_	unary minus / normalize uint8			
`	do...while	while	tic	toc
a	any	any(…, 1)	padarray / unpad array	base2base
b	bubble		strsplit	
c	char (also for cell array)	cat	strcat	strjoin / convert to '#' and char 0
d	diff	diag / spdiags	blkdiag	gcd
e	reshape / squeeze		expm / logical "infinite" graph power	exp
f	find	strfind	factor	
g	logical / cell2mat	ndgrid	gamma / gammainc / betainc	gammaln / betaln
h	horzcat	{…, …}	hankel	hypergeom
i	input	urlread	imread	
j	input(…,'s')	real	imag	conj / real and imag
k	lower / floor	upper / ceil	closest values	
l	ones	clamp (limit to a range)	log. With two inputs, specifies base	log2
m	ismember	ismember(…,'rows')	mean	lcm
n	numel / size	nchoosek (numbers) / multinomial c.	poly / interp1	
o	double / cell array to numeric / parity	int64	round / change case	fix
p	prod	prod(…, 1, ...)	cumprod	isprime / totient function
q	decrement by 1	quantile	n-th prime / next prime	primes
r	rand	randn	randi	randsample
s	sum	sum(…, 1, ...)	cumsum	std / cov / skewness / kurtosis
t	duplicate elements			strrep
u	unique	unique(,…'rows')		strjust
v	vertcat		eig / svd / strtrim	symmetric range / array / deblank
w	swap			
x	delete from stack	clc		
y	duplicate element	eye	hypot	size
z	nnz / cellfun(@nnz, ...)	nonzeros / remove whitespace		
{	*Not used. Cell array delimiter*	num2cell	mat2cell	mat2cell(x,ones(size(x,1),1),size(x,2))
|	abs / norm / determinant	union	or	bitor
}	else / finally			split array
~	Not	setxor	xor	bitxor