# Simplifying `diag` expressions*

Kevin S. Van Horn
Adobe Inc.

March 29, 2019

## 1 Definition of `diag`

We use the following definitions:

- $\mathbb{R}_0$ is the set of scalars (real numbers), $\mathbb{R}_1$ is the set of vectors over $\mathbb{R}$, $\mathbb{R}_2$ is the set of matrices over $\mathbb{R}$, and $\mathbb{R}_3$ is the set of three-dimensional arrays over $\mathbb{R}$.
- $(v \circ w)$ is the vector obtained by appending vectors $v$ and $w$, and $(s\colon v)$ is the vector obtained by prepending scalar $s$ to vector $v$.
- $(A \bullet B)$ is the block diagonal matrix whose blocks are matrices $A$ and $B$. Note that $\bullet$ is an associative operator whose left and right identity is the $0 \times 0$ matrix.
- The operation $\delta$ converts scalars, vectors, and three-dimensional arrays to matrices: for $s \in \mathbb{R}_0$, $v \in \mathbb{R}_1$, and $\alpha \in \mathbb{R}_3$, we define

  $$\begin{aligned} \delta(s) &= \text{the } 1 \times 1 \text{ matrix with single element } s \\ \delta(v) &= \text{the diagonal matrix with diagonal } v \\ \delta(\alpha) &= \alpha_1 \bullet \alpha_2 \bullet \ldots \bullet \alpha_n \end{aligned}$$

  where $n$ is the length of $\alpha$ in its first dimension.

We define the function diag as follows:

1. The domain of diag is all $n$-tuples $(x_1, \ldots, x_n)$, $n \geq 0$, such that each $x_i$ is a member of one of $\mathbb{R}_0$, $\mathbb{R}_1$, $\mathbb{R}_2$, or $\mathbb{R}_3$. The range is $\mathbb{R}_2$.

*©2019, Adobe Inc. This document is licensed to you under the Apache License, Version 2.0. You may obtain a copy of the license at http://www.apache.org/licenses/LICENSE-2.0.

2. If $n \geq 0$, $s \in \mathbb{R}_0$, $v \in \mathbb{R}_1$, $A \in \mathbb{R}_2$, and $\alpha \in \mathbb{R}_3$, then

$$\begin{aligned}
\operatorname{diag}() &= \text{the } 0 \times 0 \text{ matrix} \\
\operatorname{diag}(s, x_1, \ldots, x_n) &= \delta(x) \bullet \operatorname{diag}(x_1, \ldots, x_n) \\
\operatorname{diag}(v, x_1, \ldots, x_n) &= \delta(v) \bullet \operatorname{diag}(x_1, \ldots, x_n) \\
\operatorname{diag}(A, x_1, \ldots, x_n) &= A \bullet \operatorname{diag}(x_1, \ldots, x_n) \\
\operatorname{diag}(\alpha, x_1, \ldots, x_n) &= \delta(\alpha) \bullet \operatorname{diag}(x_1, \ldots, x_n).
\end{aligned}$$

# 2 Normal form

**Definition.** An expression $A$ is said to be *diagish* if it has the form $A = \operatorname{diag}(e_1, \ldots, e_n)$.

**Definition.** An expression $\operatorname{diag}(e_1, \ldots, e_n)$ is

- *nullary* if $n = 0$;
- *scalar-normal* if $n = 1$ and $e_1 \in \mathbb{R}_0$;
- *vector-normal* if $n = 1$, $e_1 \in \mathbb{R}_1$, and if $e_1$ is veckish it is in normal form and has at least two arguments;
- *sv-normal* if it is scalar-normal or vector-normal;
- *array-normal* if $n = 1$ and $e_1 \in \mathbb{R}_3$; and
- *unary-normal* if it is either scalar-normal, vector-normal, or array-normal.

**Definition.** An expression $A = \operatorname{diag}(e_1, \ldots, e_n)$ is *quasi-block-normal* if

- all $e_i \in \mathbb{R}_2$,
- any diagish $e_i$ is unary-normal, and
- no two consecutive $e_i$ are sv-normal diagish expressions.

$A$ is *block-normal* if it is both quasi-block-normal and $n > 1$.

**Definition.** A diagish expression is in *normal form* if it is either nullary, unary-normal, or block-normal.

# 3 Quasi-normal triples

The simplification algorithm for diagish expressions is based on the idea of a *quasi-normal triple*, which is used to scan through the arguments of a diagish expression to build a new, equivalent expression that is in normal form.

**Definition.** An expression $\operatorname{diag}(x_1, \ldots, x_n)$ is *vector-initial* if $n \geq 1$ and either $x_1 \in \mathbb{R}_0$, $x_1 \in \mathbb{R}_1$, or $x_1$ is a vector-initial diagish expression.

**Definition.** An expression $\operatorname{diag}(x_1, \ldots, x_n)$ is *vector-final* if $n \geq 1$ and either $x_n \in \mathbb{R}_0$, $x_n \in \mathbb{R}_1$, or $x_n$ is a vector-final diagish expression.

**Proposition.** *If $n > 0$ and $u = \operatorname{diag}(x_1, \ldots, x_n)$ is quasi-block-normal then $u$ is vector-final iff $x_n$ is sv-normal.*

**Definition.** A *quasi-normal triple* is a triple of diagish expressions $(A, B, C)$ such that

- $A$ is quasi-block-normal,
- all arguments of $B$ are scalars or vectors,
- if $A$ is vector-final then $B$ is nullary and $C$ is not vector-initial.

You can think of $A$ as the result arguments we have already produced, $B$ as an (eventually) sv-normal result argument we are in the process of constructing, and $C$ as the arguments we have not yet processed.

**Definition.** Two quasi-normal triples $a = (A, B, C)$ and $b = (A', B', C')$ are *equivalent*, written $a \equiv b$, if

$$A \bullet B \bullet C = A' \bullet B' \bullet C'.$$

Similarly, quasi-normal triple $(A, B, C)$ is equivalent to expression $e$ if $e = A \bullet B \bullet C$.

In order to show that our algorithm terminates, we'll need to show that some "size" metric decreases at each iteration.

**Definition.** The *diag-size* of an expression is defined by

$$\operatorname{dsize}(e) = 1 \quad \text{if } e \text{ is not diagish}$$
$$\operatorname{dsize}(\operatorname{diag}(x_1, \ldots, x_n)) = 1 + \sum_{i=1}^{n} \operatorname{dsize}(x_1)$$

**Definition.** The *QNT-size* of a quasi-normal triple is defined by

$$\operatorname{qntsize}(A, B, C) = 2 \cdot \operatorname{dsize}(C) + \begin{cases} 0 & \text{if } B \text{ is nullary} \\ 1 & \text{otherwise} \end{cases}$$

**Definition.** $a \rightarrowtail b$ ("$a$ reduces to $b$") means that if $a$ is a quasi-normal triple, then

- $b$ is a quasi-normal triple,
- $b \equiv a$, and
- $\operatorname{qntsize}(b) < \operatorname{qntsize}(a)$.

Our algorithm works by successively reducing a quasi-normal triple until no further reduction is possible. We make use of the algorithm for simplifying veckish expression.

**Definition.** If $e$ is a veckish expression then $\mathrm{simpv}(e)$ is the result of applying to $e$ the algorithm described in "Simplifying `vec` expressions." (Note that $\mathrm{simpv}(e)$ is in veckish normal form if it is veckish.)

Here are the reductions we use.

**Proposition 1.** *The following reduction properties hold:*

1. *If $y'$ is a scalar or vector expression then*

$$\begin{gathered}(A,\, \mathrm{diag}\,(y_1,\ldots,y_m)\,,\, \mathrm{diag}\,(y',x_1,\ldots,x_n)) \rightarrowtail \\ (A, \mathrm{diag}\,(y_1,\ldots,y_m,y')\,, \mathrm{diag}\,(x_1,\ldots,x_n))\,.\end{gathered}$$

2. 

$$\begin{gathered}(A,\, B,\, \mathrm{diag}\,(\mathrm{diag}\,(y_1,\ldots,y_k)\,, x_1,\ldots,x_n)) \rightarrowtail \\ (A,\, B,\, \mathrm{diag}\,(y_1,\ldots,y_k,x_1,\ldots,x_n))\,.\end{gathered}$$

3. *If $n > 0$ and $w$ does not have a scalar, vector, or diagish first argument, then*

$$\begin{gathered}(\mathrm{diag}\,(y_1,\ldots,y_m)\,,\, \mathrm{diag}\,(x_1,\ldots,x_n)\,,\, C) \rightarrowtail \\ (A',\, \mathrm{diag}(),\, C)\end{gathered}$$

*where*

$$v = \mathrm{simpv}\,(\mathrm{vec}\,(x_1,\ldots,x_n))$$
$$A' = \begin{cases}\mathrm{diag}\,(y_1,\ldots,y_m) & \text{if } v \text{ is nullary} \\ \mathrm{diag}\,(y_1,\ldots,y_m,\mathrm{diag}(s)) & \text{if } v = \mathrm{vec}(s) \\ \mathrm{diag}\,(y_1,\ldots,y_m,\mathrm{diag}(v)) & \text{otherwise}\end{cases}.$$

4. *If $y$ is neither scalar nor vector nor diagish, then*

$$\begin{gathered}(\mathrm{diag}\,(y_1,\ldots,y_m)\,,\, \mathrm{diag}(),\, \mathrm{diag}\,(y,x_1,\ldots,x_n)) \rightarrowtail \\ (\mathrm{diag}\,(v_1,\ldots,v_m,y')\,,\, \mathrm{diag}(),\, \mathrm{diag}\,(x_1,\ldots,x_n))\end{gathered}$$

*where*

$$y' = \begin{cases} y & \text{if } y \in \mathbb{R}_2 \\ \mathrm{diag}(y) & \text{if } y \in \mathbb{R}_3\end{cases}.$$

We'll need to show that, on termination, we have a normal-form expression equivalent to the original. We'll use the following.

**Proposition 2.** *If $(A, B, C)$ is a quasi-normal triple that does not match any of reductions 1–4, then both $B$ and $C$ are nullary, hence $(A, B, C) \equiv A$. Furthermore,*

- *if $A$ has exactly one argument $x$, so that $A = \operatorname{diag}(x)$, then $x$ is a matrix expression that either is not diagish or is in normal form;*
- *otherwise $A$ is in normal form.*

*Proof.* As follows:

- $C$ does not have a scalar or vector first argument (no match with reduction 1).
- $C$ does not have a diagish first argument (no match with reduction 2).
- $B$ is nullary (no match with reduction 3, and $C$ has no scalar, vector, or diagish first argument).
- $C$ is nullary (no match with reduction 4, $B$ is nullary, $C$ has no scalar, vector, or diagish first argument).

If $A$ is nullary, then it is by definition in normal form. If $A$ has exactly one argument $x$, then since $A$ is quasi-block-normal, $x$ is a matrix expression that either is not diagish or is unary-normal; in the latter case $x$ is by definition in normal form. If $u$ has more than one argument then it is block-normal, and hence in normal form. □

# 4 Algorithm for simplifying diagish expressions

Here is the algorithm:

- Input: a diagish expression $e$, all of whose arguments are either scalars, vectors, matrices, or 3D arrays.
- Output: an expression $e'$ equivalent to $e$ which, if diagish, is in normal form.
- Pseudocode:

```
A := diag(); B := diag(); C := e;
while (A, B, C) matches any of reductions 1--4:
    apply a matching reduction;
if A has exactly one argument:
    return that argument;
else:
    return A;
```

From Proposition 1 we have that $(A, B, C) \equiv e$ is an invariant of the while loop, and $\mathrm{qntsize}\,(A, B, C)$ decreases at each iteration. Since $\mathrm{qntsize}\,(A, B, C)$ is by definition a positive integer, it cannot decrease indefinitely, so the loop terminates. Proposition 2 then tells us that the returned value is equivalent to $e$ and, if diagish, is in normal form.