Pensées 8

Ashley V.

March 2, 2024

Contents

1 Objective Morality

A morality is a cost function $\mathcal{L}_0$ over the set of actions α_0 which induces a prescribed action:

$$\pi_0 = \underset{\alpha_0}{\operatorname{argmin}}\ \mathcal{L}_0(\alpha_0)$$

In the case of a meta-morality $\mathcal{L}_1$ the set of actions is the set of possible moralities α_1 which induces a prescribed morality:

$$\alpha_n = \mathcal{L}_{n-1}$$

$$\pi_1 = \underset{\alpha_1}{\operatorname{argmin}}\ \mathcal{L}_1(\alpha_1)$$

For a morality α_1 to be "objective," it must be the case that it is prescribed $\alpha_1 = \pi_1$ by an "objective" meta-morality $\mathcal{L}_1 = \alpha_2$. This objectivity condition recurs thus:

$$\pi_n = \underset{\alpha_n}{\operatorname{argmin}}\ \pi_{n+1}(\alpha_n)$$

It is not possible for a moral system π_n to be justified by $\pi_{<n}$; the set of actions cannot prescribe a judgement over themselves. The same holds at any meta-moral level.

For there to exist an "objective" π_1 there would need to exist $k \in \mathbb{Z}_+$ for which there exists only one possible π_k, and thus every possible $\mathcal{L}_k$ would have to result in an identical π_k. It possible to construct a $\mathcal{L}_n$ which prescribes any element of the set α_n and thus the set of all possible π_k has the same size as the set of all possible α_k, which is clearly infinite (given $k > 0$, there may be only one possible action), as there exists an infinite number of possible cost functions. If α_k is reduced into classes based on its sorted results, then the size of the set is still monotonically and quickly increasing with k.

$$|\{\pi_n \mid \exists \mathcal{L}_n\ \pi_n = \underset{\alpha_n}{\operatorname{argmin}}\ \mathcal{L}_n(\alpha_n)\}| = |\alpha_n|$$

An "objective" morality may be suggested as a morality which justifies itself, that is, roughly, $\forall i, j \in \mathbb{Z}_+\ \mathcal{L}_i = \mathcal{L}_j$. It is valid to view $\mathcal{L}$ as segmented based on the level which it is applied to, and an identical problem arises. The "objectivity" of it applied on any level would imply it on all others, but there is no contradiction in it lacking "objectivity" on all levels.

The only solution to this problem is to select a π_k to hold as an axiomatic singleton and justifying all lower levels, demonstrating that all $\pi_{<k}$ can only be justified by axioms π_k, therefore an "objective" morality, one prescribed without the acceptance of any arbitrary axioms, does not exist.

2 RREF Algorithm

2.1 Motivation

In the general case, the RREF of a matrix will have a block which is I in the upper left, zeros below it, and then a section to the right of it which is non-trivial (although one or more may be absent). Example:

$$A = \begin{bmatrix} 5 & 1 & 4 & 2 \\ 1 & 1 & 5 & 7 \\ 9 & 1 & 2 & 6 \\ 5 & 1 & 4 & 2 \end{bmatrix} \quad \text{rref}(A) = \begin{bmatrix} 1 & 0 & 0 & -3.5 \\ 0 & 1 & 0 & 55.5 \\ 0 & 0 & 1 & -9 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The RREF can be computed by linear combinations, scalings, and permutations of rows, and thus through left-multiplication with a matrix R such that $\text{rref}(A) = RA$. The matrix R is independent of what is in the last row given that the rank is 3, suggesting the last column is 0's, and the resulting last row is zero, suggesting the last row of R be as well. You may check that $RA = \text{rref}(A)$.

$$R = \begin{bmatrix} 0.75 & -0.5 & -0.25 & 0 \\ -10.75 & 6.5 & 5.25 & 0 \\ 2 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$\text{rank}(A) = 3$, the 3×3 block is found to be the inverse of the upper left 3×3 block of A. This suggests that the RREF can be computed by determination or ahead of time knowledge of the rank, then ordinary matrix inversion and multiplication, along with adequate padding. In addition, a large subset of possible matrices have RREFs which can be fully determined only from their shape and rank, enabling faster computation.

2.2 Outline

$$A \in \mathbb{R}^{m \times n} \quad r = \text{rank}(A) \quad \text{rref}(A) = RA \in \mathbb{R}^{m \times n}$$

Case 0: The RREF of a full-rank square matrix is trivially the identity matrix and thus we can return early.

$$r = n = m \Rightarrow RA = I$$

Case 1: The RREF of a matrix with extra rows but the same rank as number of columns is trivially the RREF of the matrix with those rows removed, padded with zeros to restore them.

$$r = n < m \Rightarrow RA = \begin{bmatrix} A_0^{-1} A_0 \\ 0 \end{bmatrix} = \begin{bmatrix} I \\ 0 \end{bmatrix}$$

Case 2: If we have the same rank as the number of rows we split it up the same way but the extra columns are linear combinations of the other columns, we solve $A_0 X = A_1$, $X = A_0^{-1} A_1$ to find what those are.

$$r = m < n \Rightarrow RA = \begin{bmatrix} A_0^{-1} A_0 & A_0^{-1} A_1 \end{bmatrix} = \begin{bmatrix} I & A_0^{-1} A_1 \end{bmatrix}$$

Case 3: In the worst case we have both extra rows and columns, we combine these methods.

$$r < n, r < m \Rightarrow RA = \begin{bmatrix} A_0^{-1} A_0 & A_0^{-1} A_1 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} I & A_0^{-1} A_1 \\ 0 & 0 \end{bmatrix}$$

2.3 Pseudocode

Algorithm 1: RREF

Data: $A \in \mathbb{R}^{m \times n}$
Result: $\text{rref}(A) \in \mathbb{R}^{m \times n}$

$A = \text{QR}(A).\text{R}$ // $A = QR,\ A := R$
$r = \text{CountNonzero}(\text{Diag}(R))$ // Get rank
if $r = n$ **then** // Case 0 or 1
  Return $\text{Eye}(m, n)$ // Not necessarily square
$A[:\text{r}, \text{r}:] = A[:\text{r}, :\text{r}]^{-1} * \text{A}[:\text{r}, \text{r}:]$ // $A_1 := A_0^{-1} A_1$
$A[:\text{r}, :\text{r}] = \text{Eye}(r)$ // $A_0 := I$
if $r < m$ **then** // Case 3
  $\text{A} = \text{PadRows}(\text{A}, m)$ // Add zero rows back
Return A

2.4 Python Code

```
import numpy as np

def rref(a: np.ndarray) -> np.ndarray:
    m, n = a.shape

    a = np.linalg.qr(a, mode="r")

    r = np.count_nonzero(~np.isclose(np.diag(a), 0))

    if r == n:
        return np.eye(m, n)

    a[:r, r:] = np.linalg.inv(a[:r, :r]) @ a[:r, r:]
    a[:r, :r] = np.eye(r)

    if r < m:
        a = np.vstack([a, np.zeros([m - a.shape[-2], n])])

    return a
```

2.5 Conclusion

It is possible that computing the rank via QR decomposition is less numerically stable than via singular values, but it is significantly faster. The only part of the output dependent on the elements of A is $A_0^{-1} A_1$, which is only present given $r \neq n$. Knowledge of the rank of the matrix beforehand or a more efficient means of computation would significantly reduce the computation time.

This method of computing the RREF is significantly faster and likely more numerically stable than any others of which I'm aware, especially with knowledge of the rank, although there is an obvious dearth of effort put into optimizing it because of its very limited applications (I know of no real applications outside of a classroom).