Implicit Regularization in Matrix Factorization

Wiktor Jurasz

Seminar: Optimization and Generalization in Deep Learning

Abstract

In this report we take a closer look at gradient descent optimization of an underdetermined quadratic objective over a matrix W. We examine direct optimization of W as well as its factorization. We show, that under special conditions, such optimization yields frobenius norm for direct W optimization, and nuclear norm when preforming the descent on W factorization. This is not a research paper and credits for discovery of this phenomena go to Gunasekar et al. [3]. Our work focus on providing intuitive understanding as well as more detailed formal proof.

1 Introduction

Underdetermined optimization problems have multiple global optimas which despite all being exact answers to the problem might yield very different generalization properties. To better understand the problem, one might think about fitting n degree polynomial using less than n data points (Figure 1). There is infinitely many values for coefficients of such polynomial which fit the data perfectly, however just by plotting the result it became obvious that many of them very poorly describe underlying distribution of the data. To force the gradient descent optimizer to obtain more general and "simpler" solutions one can change the objective by adding norm regularization term. In this way, the optimizer is penalized when choosing very complex (e.g. high degree) optima, which leads to more smooth solutions that generalize better.

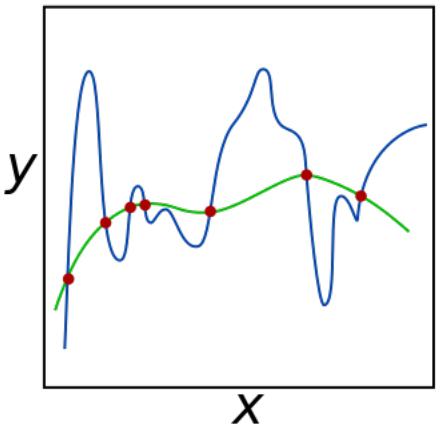

Figure 1: Fitting high degree polynomial. Blue line represents "complex" solution that generalize badly, the green one is regularized "simple" solution.

However, recent studies [5][4] show, that such underdetermined systems (e.g. some deep models) generalize well even without explicit regularization terms. When models are trained by minimizing training error of plain objective function (without any penalizing term), there seems to be nothing preventing them from obtaining some global optima that generalizes very poorly. Yet, the gradient descent based optimisers seem to prefer certain solutions over the others. So, there should be some implicit regularization measures biasing the optimisation towards "simpler" models.

The question "why" gradient descent prefers better solutions has yet to be answered. However, recent research has shed the light on the question "which" solutions are those. In this report we take a closer look at optimisation of an underdetermined quadratic objective over a matrix W. We show that solutions chosen by optimizer correspond to minimum norm solutions. We also discuss how

different setups of optimization, lead to different norms. Finally, we give intuitive understanding of why minimum norm solutions are "simpler" and correspond to better generalization.

2 Motivation

Before examining the implicit regularization properties of gradient descent optimisation over a single matrix, let's try to place the problem in broad field of deep learning.

$$E(W) = \frac{1}{2}\sum_{i=1}^{m} \|f(x_i, W) - y_i\|^2 \tag{1}$$

Equation 1 shows an example cost function which we want to minimize, where f is a neural network. The f function can be extremely complex containing normalization, convolutions, recurrent connections, etc. It makes reasoning about neural networks in general very difficult.

$$f(x_i, W) = W_m\sigma_n(W_{m-1}\sigma_{m-1}(\dots\sigma_0(W_0x_i)\dots)) \tag{2}$$

Instead let's take a simple feed forward network, which can be seen as in Equation 2. But even in this case, the nonlinearities σ_i introduce a lot of complexity, and since the goal is to explore some basic behaviours further simplification is necessary.

$$f(x_i, W) = W_mW_{m-1}\dots W_0x_i = Wx_i \tag{3}$$

After removing nonlinearities the network is just chain of matrix multiplication, which can be reduced further to matrix vector multiplication.

$$E(W) = \frac{1}{2}\sum_{i=1}^{m}(Wx_i - y_i)^2 \tag{4}$$

Simplifying it further by setting $y_i, x_i \in \mathbb{R}$, we obtain cost function as in Equation 4.

$$F(W) = \frac{1}{2}\sum_{i=1}^{m}(\langle A_i, W\rangle - y_i)^2 = \frac{1}{2}\sum_{i=1}^{m}(\sum_{j,k} W_{jk}A_{ijk} - y_i)^2 \tag{5}$$

Finally, we modify the function E by replacing vector multiplication with matrix scalar product. In this way we obtain the function F, the same as studied in Gunasekar et al. [3] paper. The choice of such function F is somehow arbitrary, but as shown above, it is closely related to the cost function of very simple neural network. Also, as we explain in next section, F can be seen as matrix recovery problem.

It is important to emphasise that we do not directly study implicit regularization in neural networks on matrix recovery. Function F is just ***an example*** function, which allow as to examine the regularization effect for some matrix W. This can be viewed as base step for further research for more general and complex systems.

3 Stating Optimization Objective

So far we have only stated function F, but now we have to phrase the optimization problem. It comes handy to interpret optimization of F in Equation 5 as matrix recovery problem.

3.1 Matrix Recovery Intuition

For matrix recovery we have:

- Input: Matrix W with missing entries.
- Goal: Recover full matrix W (i.e. find missing entries)

Now, to fit this into the Equation 5 we set:

- $i = 1 \dots m$, where m is the number of known entries
- $A_i \in \mathbb{R}^{n \times n}$ are the matrices of indicators, consisting of all 0 and one 1 (1 indicates the entry of W that is known).

$$A_i = \begin{bmatrix} 0 & 0 & \dots & 0 \\ \dots & & 1 & . \\ 0 & 0 & \dots & 0 \end{bmatrix}$$

- $y_i \in \mathbb{R}$ is the value of known entry indicated by matrix A_i.
- $W \in \mathbb{R}^{n \times n}$ is the matrix to recover that is completely erased and reinitialized (since all the information about known entries was transfered to A_i and y)

3.2 Constrain Optimization Formulation

With the assumption that $m \ll n^2$ we can state underdetermined optimization objective:

$$\min_{W} \quad F(W) = \frac{1}{2} \sum_{i=1}^{m} (\langle A_i, W \rangle - y_i)^2 \tag{6}$$

However, without constraints we are able to minimize such problem trivially. For matrix recovery, just by setting back the known entries of W and setting some arbitrary values in other fields.

$$\min_{W \succcurlyeq 0} \quad F(W) = \frac{1}{2} \sum_{i=1}^{m} (\langle A_i, W \rangle - y_i)^2 \tag{7}$$

Instead we are going to work on contrained optimization where W has to be positive-semidefinite (PSD), which limits number of trivial solutions. This is again somewhat arbitrary choice, but PSD matrices have some nice properties (e.g. real, positive eigenvalues). PSD is also often required in different "real" optimization problems.

At the beggining we said, that "simpler" solutions yield better generalization. But what does "simple" mean in context of matrix? One answer is *rank*.

Definition 1. ***Rank*** *of a matrix A is the dimension of the vector space generated (or spanned) by its columns. This corresponds to the* ***maximal number of linearly independent columns of A****.*

$$A \in \mathbb{R}^{n \times m} \Rightarrow rank(A) \leq min(n, m) \tag{8}$$

$$A = BC \Rightarrow rank(A) \leq min(rank(B), rank(C)) \tag{9}$$

Think about rating matrix, where columns are users, rows are products and entries are rates that the users gave to the products. For majority of real-world retailers such matrix is very sparse ($m \ll n^2$) and also very big (number of products and customers go into millions). But, in reality there is relatively few features all customers/products have that drive the ratings. So the real recovered matrix would be low rank.

$$\min_{UU^T = W \succcurlyeq 0} \quad f(U) = \frac{1}{2} \sum_{i=1}^{m} (\langle A_i, UU^T \rangle - y_i)^2 \tag{10}$$

Now, using properties in Equation 8 and Equation 9 we can impose low rankness of matrix W by setting $W = UU^T$ where $U \in \mathbb{R}^{n \times d}$ and manipulating the d. However, for neural networks (we can see UU^T as simple 2-layer network), there are no explicit dimension restrictions (layers can be arbitrary wide) that would guarantee low rank (thus "simple") solution. So the question we should ask is what happens for any d (and for $d \geq n$ in particular). More formally we can consider if minimizing rank of W (Equation 11) is somehow related to performing gradient descent on $W = UU^T$ in Equation 10?

$$\begin{aligned} \min_{W \succcurlyeq 0} \quad & rank(W) \\ \text{s.t.} \quad & \langle A_i, X \rangle = y_i \quad \forall i = 1 \dots m \end{aligned} \tag{11}$$

3.3 Non-convexity of Rank

The problem with the rank minimization is the fact that it is not a convex function. Luckily for us, it was proven [2] that minimizing nuclear norm is equivalent to minimizing convex envelope of rank. Thus, we can replace rank in Equation 11 to obtain convex optimization problem, which is much easier to work with.

$$\begin{aligned} \min_{W \succcurlyeq 0} \quad & \|W\|_* \\ \text{s.t.} \quad & \langle A_i, X \rangle = y_i \quad \forall i = 1 \dots m \end{aligned} \tag{12}$$

Definition 2. ***Nuclear norm*** *is a sum of all singular values of a matrix.*

$$\|W\|_* = \sum_{i=1}^{min(n,m)} \sigma_i(W)$$

Definition 3. ***Rank*** *of a matrix W is the number of non zero singular values (σ_i) of this matrix.*

However, just by looking at the definition of the nuclear norm it is not immediately clear why minimizing it also minimizes the rank. Since the rank is equal the number of non-zero singular values, why just by minimizing their sum we obtain sparse singular values vector?

To gain some intuition about it, take a look at the Figure 2. For some constrained minimization of nuclear norm, the minimum value is c, i.e. $\|X^*\|_* = c$. The optimization constraint (for example our $\langle A_i, X \rangle = y_i$) also must be satisfied in this point (Here we are mixing space of X and space of σ, but since σ is a function of X, the constrain on X also affect σ). As we can see almost always those constraints are satisfied in one of pointy corners of the sigma space. Because of this "pointy" characteristic, the minimum value of the nuclear norm is most likely obtained at one of the corners of the nuclear norm n-dimensional ball, which yields sparse vector of singular values, which means low rank solution.

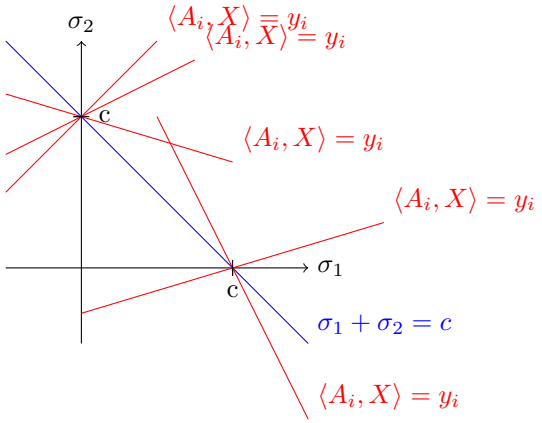

Figure 2: Most of the constrain functions cross point $(c, 0)$ or $(0, c)$, making the singular values vector sparse.

4 Implicit Regularization in Matrix Factorization

We are interested in behaviour of gradient descent when optimizing previously stated underdetermined systems over matrix W and its factorization. Namely, we want to examine implicit regularization behaviour over W in following objectives:

$$\min_{W} \quad F(W) = \frac{1}{2} \sum_{i=1}^{m} (\langle A_i, W \rangle - y_i)^2 \tag{13}$$

and

$$\min_{UU^T=W\succcurlyeq 0} \quad f(U) = \frac{1}{2}\sum_{i=1}^{m}(\langle A_i, UU^T\rangle - y_i)^2 \tag{14}$$

And show that optimal gradient descent solution (i.e. when training error goes to zero), corresponds to minimum frobenius norm for Equation 13 and minimum nuclear norm for Equation 14.

4.1 Gradient Flow

First step is to derive gradient flow, with infinitely small step, for both of those objectives. For simplicity, all the derivations are done for $m = 1$ and $A_1 = A$, $y_1 = y$.

$$\begin{aligned} W_t &= W_{t-1} - \lim_{\tau\to 0}\tau\nabla F(W_{t-1}) \\ \dot{W}_t &= \frac{W_t - W_{t-1}}{\lim_{\tau\to 0}\tau} = \frac{dW_t}{dt} = -\nabla F(W_{t-1}) \end{aligned} \tag{15}$$

Hence, we have that is infinitely small change in W is equal to $-\nabla F(W)$.

$$\nabla F(W_{t-1}) = \nabla\frac{1}{2}(\langle A, W_{t-1}\rangle - y)^2 = \nabla\frac{1}{2}(Tr(A^T W_{t-1}) - y)^2 = (\langle A, W_{t-1}\rangle - y)A \tag{16}$$

In similar way, we can obtain gradient flow for U:

$$\dot{U}_t = \frac{U_t - U_{t-1}}{\lim_{\tau\to 0}\tau} = \frac{dU_t}{dt} = -\nabla f(U_{t-1}) = -\nabla\frac{1}{2}(\langle A, U_{t-1}U_{t-1}^T\rangle - y)^2 = -(\langle A, U_{t-1}U_{t-1}^T\rangle - y)AU_{t-1} \tag{17}$$

However we are not really interested in gradient flow on U. Our goal is to find W, so we must ask **What is the gradient flow on W when gradient descent is done on its factorization U.** Because $W_t = U_t U_t^T$, by chain rule:

$$\begin{aligned} \dot{W}_t = \dot{U}_t U_t^T + U_t\dot{U}_t^T = -(\langle A, U_t U_t^T\rangle - y)AU_t U_t^T - U_t U_t^T(\langle A, U_t U_t^T\rangle - y)A = \\ -(\langle A, W_t\rangle - y)AW_t - W_t(\langle A, W_t\rangle - y)A = -r_t AW_t - W_t r_t A \end{aligned} \tag{18}$$

Where $r_t = \langle A, W_t\rangle - y$.

It is important to realize that we obtained two different gradient flows for W. $\dot{W}_t = -r_t A$ when working directly on W and $\dot{W}_t = -r_t AW_t - W_t r_t A$ when working on its factorization.

4.2 Proof Strategy

To show that those gradient flows are equivalent to minimizing norm objectives, we use Karush–Kuhn–Tucker (KKT) and Slater's conditions. Those are part of standard optimization theory and in-depth description can be find in any good convex optimization textbook.

The main arch of the proof relays on fact that if the Slater's conditions are satisfied (which is the case for our norm minimization problems) then KKT conditions become *sufficient* to obtain global minimum of the optimization objective. In other words, under Slater's conditions it's enough to show, that some argument satisfy the KKT conditions, to conclude that this argument is indeed a global minimizer.

Following this logic we show, that under some assumptions matrix $\hat{W}$ obtained from gradient descent satisfy KKT conditions of norm minimization objective.

4.3 Gradient Descent on W

For frobenius norm minimization problem, i.e. $\min_W \|W\|_2^2, s.t. \quad \langle A, W\rangle = y$ we have following KKT conditions coming from primal feasibility and Lagrangian: $\langle A, W\rangle = y$ and $\exists v \in \mathbb{R}, s.t. \quad W = vA$.

The first one is satisfied if during descent we converge to zero error (which most likely is the case for underdetermined system). To satisfy the second one we need a following assumption: $W_0 = \lim_{\alpha\to 0}\alpha I$. Then indeed we can always find v that satisfy the Lagrangian condition.

To see this consider first gradient update step: $W_1 = W_0 - \lim_{\tau\to 0}\tau\nabla F(W_0) = \lim_{\alpha\to 0}\alpha I - \lim_{\tau\to 0}\tau\nabla\frac{1}{2}(\langle A, \lim_{\alpha\to 0}\alpha I\rangle - y)^2$. In such setup W will always be a scaled A as we never leave the subspace of A. From this and the fact that the Slater's conditions are satisfied we can conclude that gradient descent over W in F yields minimum frobenius norm solution.

4.4 Gradient Descent on $W = UU^T$

Theorem 1. *For gradient flow $\dot{W}_t = -r_t A W_t - W_t r_t A$ and following assumptions:*

$$\begin{aligned} W_0 &= \alpha I \\ W_t &= W_{t-1} - \lim_{\tau\to 0}\tau\nabla f(W_{t-1}) \\ W_\infty &= \lim_{t\to\infty} W_t(\alpha I) \\ \hat{W} &= \lim_{\alpha\to 0} W_\infty(\alpha I) \end{aligned} \tag{19}$$

If $\hat{W}$ exists and is global optimum for Equation 14, then it is also an optimal solution for the following problem for any diagonalizable matrix A:

$$\begin{aligned} \min_{W\succcurlyeq 0} \quad & \|W\|_* \\ s.t. \quad & \langle A, W\rangle = y \end{aligned} \tag{20}$$

KKT conditions for Equation 20 are as follows:
$\exists v \in \mathbb{R}, s.t. \quad (1)\, \langle A, W\rangle = y, \quad (2)\, W \succcurlyeq 0, \quad (3)\, vA \preccurlyeq I, \quad (4)\, W = vAW$

Equation 20 is a semidefinite programming task, which is a subset of convex optimization field. Thus, to o understand why the KKT conditions are in this form one can again use standard convex optimization thoery textbook.

Proof sketch: Condition (1) is satisfied if (as in previous case) gradient descent converges to zero. Condition (2) can be proved using SVD, i.e. $W = UU^T = U\Sigma V^T V\Sigma U^T = U\Sigma^2 U^T \succcurlyeq 0$. Conditions (3) and (4) are much harder to prove and intuitively understand. We can show that under the assumptions and correctly picked v, vA will have eigenvalues bounded by 1 and vAW will have the same eigenvalue and eigenvectors as W, thus the conditions will be met. We include formal derivation in appendinx B.

5 Summary

5.1 Generalization of the proof

The above derivations can be generalized for arbitrary m, not diagonalizable matrix A and for asymmetric factorization (i.e. $W = UV^T$). This is shown in Gunasekar et al. [3] paper.

5.2 Conclusion

The result of this research prove, that there exists linkage between matrix norm and gradient descent optimization. It also shows, that factorization of the matrix (somehow similar to introducing depth in neural network), changes the global minimizer. The simplicity of minimum nuclear norm solution has its reflection in the success of gradient descent optimizers. However, because of very restrictive conditions like: infinitely small initialization and step size, infinitely many steps and linear 2-factorization, its still theoretical-only result. It is nevertheless an important step in "understanding" of deep learning, that is already a base for more general research[1].

In this report we provided an intuitive understanding of why and how implicit regularization in matrix factorization appears in modern deep learning field. We showed how narrowing down the scope by considering simpler cases can allow one to reason about unknown phenomena. Finally, we examined findings of Gunasekar et al. [3], explained their proof strategy and provided formal and intuitive understanding of their research.

References

[1] Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization, 2019.

[2] Maryam Fazel. *Matrix rank minimization with applications*. PhD thesis, Stanford University, 2002.

[3] Suriya Gunasekar, Blake Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nathan Srebro. Implicit regularization in matrix factorization, 2017.

[4] Behnam Neyshabur, Ryota Tomioka, Ruslan Salakhutdinov, and Nathan Srebro. Geometry of optimization and implicit regularization in deep learning, 2017.

[5] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning, 2014.

Appendices

A Lemmas

A.1 Lemma 1

Let $A, B \in \mathbb{R}^{n\times n}$ and v be an eigenvector of both A and B. Moreover let α be an eigenvalue of A corresponding to eigenvector v and β be an eigenvalue of B corresponding to eigenvector v.

Then v is also an eigenvector of AB with corresponding eigenvalue $\alpha\beta$

Proof: $Av = \alpha v$ and $Bv = \beta v$
$ABv = A(\beta v) = \beta Av = \beta\alpha v$ □

B Theorem 1 proof

We have already proved primal feasibility (i.e. condition (1) and (2)). Here we provide formal proof of dual feasibility (3) and complementary slackness (4).

Preparation: First we can show that if gradient flow $\dot{W}_t = -r_t(AW_t + W_tA)$, then

$$W_t = exp(-s_tA)W_0exp(-s_tA) \tag{21}$$

where $s_t = -\int_0^T r_t dt$ and $r_t = \langle A, W_t\rangle - y$ and $exp(X) = \sum_{i=0}^{\infty}\frac{X^i}{i!}$ is a matrix exponential.

To see this, set: $g_1(s_t) = exp(-s_tA)W_0$ and $g_2(s_t) = exp(-s_tA)$ and $W_t = g_1(s_t) * g_2(s_t)$

$$\begin{aligned}
&\frac{dW_t}{dt} = \frac{d(g_1(s_t) * g_2(s_t))}{dt} = \frac{d(g_1(s_t))}{dt}g_2(s_t) + g_1(s_t)\frac{d(g_2(s_t))}{dt} = \\
&\quad \frac{d(exp(s_tA)W_0)}{dt}exp(s_tA) + exp(s_tA)W_0\frac{d(exp(s_tA))}{dt} = \\
&\frac{d(s_tA)}{dt}exp(s_tA)W_0exp(s_tA) + exp(s_tA)W_0exp(s_tA\frac{d(s_tA)}{dt} = \\
&\quad \frac{d(-\int_0^T r_t dt)}{dt}(AW_t + W_tA) = -r_t(AW_t + W_tA) = \dot{W}_t
\end{aligned} \tag{22}$$

Because $A \in \mathbb{R}^{n\times n}$ is diagonalizable is has an eigenbasis $e_1, \ldots, e_n$ such that all the eigenvectors are linearly independent. Matrix exponentiation and scalar multiplication doesn't change the eigenbasis, so $exp(-s_tA)$ has the same eigenbasis. Now, if we rewrite Equation 21 using initialization assumptions we obtain:

$$W_t = exp(2s_tA)U_0U_0^T = exp(2s_tA)\lim_{\alpha\to 0}\alpha^2 I = exp(2s_tA + 2\ln\alpha I) \tag{23}$$

We can conclude, that W_t and $\hat{W}$ also has the same eigenbasis.

Proof that (3) **holds:** For W_∞ and $\beta = -\ln \alpha I$ Equation 23 takes following form: $W_\infty = exp(2s_\infty(\beta)A - 2\beta)$. Now,

$$\lambda_k(W_\infty) = \lambda_k(exp(2s_\infty(\beta)A - 2\beta)) = exp(\lambda_k(2s_\infty(\beta)A) - 2\beta) \tag{24}$$

Where $\lambda_k(W_\infty)$ is k-th eigenvalue of W_∞.

For $\hat{W}$ β goes to infinity, so by rearranging Equation 24 and taking ln from both sides, we obtain for $\lambda(\hat{W}) \neq 0$ (which imply $\lambda(\hat{W}) > 0$ because of factorization):

$$\lambda_k(2s_\infty(\beta)A) - 2\beta - \ln \lambda_k(\hat{W}) \to 0 \Rightarrow \lambda_k(\frac{s_\infty(\beta)}{\beta}A) - 1 - \frac{\ln \lambda_k(\hat{W})}{2\beta} \to 0 \tag{25}$$

From this we conclude:

$$\lim_{\beta \to \infty} \lambda_k(\frac{s_\infty(\beta)}{\beta}A) = 1 \tag{26}$$

For the case when $\lambda(\hat{W}) = 0$ we have:

$$exp(\lambda_k(2s_\infty(\beta)A) - 2\beta) = exp(2\beta(\lambda_k(\frac{s_\infty(\beta)}{\beta}A) - 1)) \to 0 \tag{27}$$

If $\beta \to \infty$ then for whole expression to be true it must hold that:

$$\lim_{\beta \to \infty} \lambda_k(\frac{s_\infty(\beta)}{\beta}A) < 1 \tag{28}$$

From Equation 26 and Equation 28 we have that for $v = \lim_{\beta \to \infty} \frac{s_\infty(\beta)}{\beta}$ (3) holds. □

Proof that (4) **holds:** Let $\hat{v} = \lim_{\beta \to \infty} \frac{s_\infty(\beta)}{\beta}$ as we showed before $\hat{v}A$ and $\hat{W}$ have the same eigenvectors. From *Lemma 1* and from previous proof of (3) we have that

$$\lambda_k(\hat{v}A\hat{W}) = \lambda_k(\hat{v}A)\lambda_k(\hat{W}) = \lambda_k(\hat{W}) \tag{29}$$

Now, again from *Lemma 1* eigenvectors of $\hat{W}$ are the same as $\hat{v}A\hat{W}$. This together with the fact that A is diagonalizable allow as to conclude that there exists matrix P with eigenvectors $e_1 \dots e_n$ as columns such that:

$$\begin{aligned} P^{-1}\hat{W}P &= \Sigma_1 \\ P^{-1}\hat{v}A\hat{W}P &= \Sigma_2 \end{aligned} \tag{30}$$

From Equation 29

$$\Sigma_1 = \Sigma_2 \tag{31}$$

and finally:

$$P^{-1}\hat{W}P = P^{-1}\hat{v}A\hat{W}P \Rightarrow \hat{W} = \hat{v}A\hat{W} \tag{32}$$

which concludes that for $\hat{v}$ (4) holds. □