Selforganizing Clifford Neural Network

Eduardo Bayro-Corrochano, Sven Buchholz, Gerald Sommer
Computer Science Institute, Christian Albrechts University, Kiel, Germany.
email: edb, sbh, gs@informatik.uni-kiel.d400.de

Abstract

This paper presents a novel selforganizing type RBF neural network and introduces the geometric algebra in the neural computing field. Real valued neural nets for function approximation require feature enhancement, dilation and rotation operations and are limited by the Euclidean metric. This coordinate-free geometric framework allows to process patterns between layers in a particular dimension and desired metric being possible only due to the promising projective split. The potential of such nets working in a Clifford algebra $\mathcal{C}(V_{p,q})$ is shown by a simple application of frame coordination in robotics.

1 Introduction

Geometric algebra is a coordinate-free approach to geometry based on the algebras of Grassmann and Clifford and has already been successfully applied to many areas of mathematical physics and engineering. The approach we adopt here is due to Hestenes [1]. A very brief outline of the algebra will be given ([1-2] for more details) . For an n-dimensional space it can be introduced an orthonormal basis of vectors $\{\sigma_i\}$ $i = 1, ..., n$ such that $\sigma_i \cdot \sigma_j = \delta_{ij}$. This leads to a basis for the entire Clifford algebra $\mathcal{C}(V_{p,q})$ (p even and q odd elements):

$$1, \quad \{\sigma_i\}, \quad \{\sigma_i \wedge \sigma_j\}, \quad \{\sigma_i \wedge \sigma_j \wedge \sigma_k\}, \quad \ldots, \quad \sigma_1 \wedge \sigma_2 \wedge \ldots \wedge \sigma_n.$$

The highest grade element is called the *pseudoscalar* of the space. Any multivector can be expressed in terms of this basis. As well as vector addition and scalar multiplication we have a non-commutative product which is associative and distributive over addition – this is the **geometric** or **Clifford** product. The geometric product of two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ can be expressed as a sum of its symmetric and antisymmetric parts $\boldsymbol{ab} = \boldsymbol{a}\cdot\boldsymbol{b} + \boldsymbol{a}\wedge\boldsymbol{b}$. The inner product $\boldsymbol{a}\cdot\boldsymbol{b}$ is the standard scalar product and produces a scalar. The outer or wedge product $\boldsymbol{a}\wedge\boldsymbol{b}$ is a new quantity we call a *bivector* – the directed area formed by sweeping $\boldsymbol{a}$ along $\boldsymbol{b}$. $\boldsymbol{b}\wedge\boldsymbol{a}$ will have the opposite orientation. This is immediately generalizable to higher dimensions, e.g. $(\boldsymbol{a}\wedge\boldsymbol{b})\wedge\boldsymbol{c}$ is a **trivector** (oriented volume). The outer product of k vectors is a k-vector or k-blade, and is said to have *grade* k. A **multivector** is any linear combination of different grade objects. We can easily extend the above definition of the geometric product to enable us to multiply any two blades and thus to multiply any two multivectors. The outer product is much more useful than the standard cross product which is not generalizable to dimensions other than 3. The 2^3-dimensional *Pauli-Algebra* has the following basis: 1 (scalar), $\sigma_1, \sigma_2, \sigma_3$ (vectors), $\sigma_1\sigma_2, \sigma_2\sigma_3, \sigma_3\sigma_1$ (bivectors), $\sigma_1\sigma_2\sigma_3 \equiv i$ (trivector). By straightforward multiplication it can be easily seen that the three bivectors can also be written as $\sigma_2\sigma_3 = i\sigma_1 = \boldsymbol{i}$, $\sigma_1\sigma_3 = -i\sigma_2 = \boldsymbol{j}$, $\sigma_1\sigma_2 = i\sigma_3 = \boldsymbol{k}$. These simple bivectors are spinors and using them can be proved that the quaternion algebra of Hamilton is a subset of the geometric algebra of space. In the 3-D space we use the term 'rotor' for those even elements of the space that represent rotations. Any rotor can be written in the form $\mathbf{R} = \pm e^{\mathbf{B}/2}$, where $\mathbf{B}$ is a bivector. In particular, in 3-D we write $\mathbf{R} = e^{(-i\frac{\theta}{2}\mathbf{n})} = \cos\frac{\theta}{2} - i\mathbf{n}\sin\frac{\theta}{2}$ which represents a rotation of θ radians anticlockwise about an axis parallel to the unit vector $\mathbf{n}$. A potentially very useful expression for the rotation operator $\mathbf{R}$ of a m-dimensional multivector $\mathbf{x}$ is $\mathbf{y} = \mathbf{R}\mathbf{x}\tilde{\mathbf{R}} = \mathbf{e}^{-\mathbf{B}/2}\mathbf{x}\mathbf{e}^{\mathbf{B}/2}$ where now $\mathbf{B}$ is a m-bladed bivector. This can be further decomposed into a sequence of rotations by angles $|\theta_{2k}|$

in particular i_{2k}-planes $\mathbf{y} = \mathbf{R_m R_{m-1} ... R_1 x \tilde{R}_1 ... \tilde{R}_{m-1} \tilde{R}_m}$. where $\mathbf{R_k} = \mathbf{e^{-B_{2k}/2}}$ and $\mathbf{\tilde{R}_k} = \mathbf{e^{B_{2k}/2}}$.

2 Real Valued Network Structures for Approximation

The approximation of general continuous functions using nonlinear networks is useful in areas of system modelling and identification. Cybenko [3] used for the approximation the superposition of weighted sigmoidal functions as follows. If $\sigma(.)$ is a continuous discriminatory function like a sigmoidal, then a finite sum of the form

$$g(\boldsymbol{x}) = \sum_{i=1}^{N} w_{2i} \sigma[\boldsymbol{w}_{1i}^{T}(\boldsymbol{x} - \boldsymbol{t}_i)] \tag{1}$$

will approximate a given continuous function *f*, where $w_{2i} \in \mathcal{R}$ and $\boldsymbol{x}$; $\boldsymbol{t}_i, \boldsymbol{w}_{1i} \in \mathcal{R}^n$. For a given $\varepsilon > 0$, there is a sum given by equation (1), for which $|g(\boldsymbol{x}) - f(\boldsymbol{x})| < \varepsilon$ for all $\boldsymbol{x} \in [0,1]^n$. Poggio and Girosi [4] used instead for the same purpose a superposition of weighted Gaussian functions

$$g(\boldsymbol{x}) = \sum_{i=1}^{N} w_{2i} G[\mathtt{D}_i(\boldsymbol{x} - \boldsymbol{t}_{1i})] \tag{2}$$

where $w_{2i} \in \mathcal{R}$, G is a Gaussian function, $\mathtt{D}_i$ a $N \times N$ dilation diagonal matrix and $\mathbf{x}, \mathbf{t}_{1i} \in \mathcal{R}^n$. The vector $\mathbf{t}_{1i}$ is a translation vector. These nets use real values for signal representation and processing and are unfortunately limited by the Euclidean metric. In order to improve the representation and processing of signals the authors designed a neural network in the geometric algebra framework. Patterns are represented as multivectors and the linear correlator is replaced by a geometric correlator.

3 Clifford Algebra Network Structures for Approximation

Let us analyze each case of section 2 in terms of this new geometric processing. Note that $\boldsymbol{w}_{2i}$, $\boldsymbol{w}_{1i}$, $\boldsymbol{x}$, $\boldsymbol{t}_i$ are now multivectors and the geometric correlator is used instead the scalar correlator. Since Clifford Algebra is not in general a division algebra, it is not possible to define higher dimensional analogues of the sigmoid function [5]. In the first case, equation (1), the activation function proposed by Georgiou et al [6] is used instead of the sigmoidal function. This complex activation function is bounded and nonlinear and its partial derivative exists and is continuous

$$\mathbf{u}(\mathbf{z}) = \frac{\mathbf{z}}{c + \frac{1}{r}|\mathbf{z}|} \tag{3}$$

where $\mathbf{z}$ is any multivector. The inner vector product will be extended to a geometric product in order to enhance the features for a better processing, namely
$\mathbf{w}_{1i}^T(\mathbf{x}-t_i) \Rightarrow \mathbf{w}_{1i}(\mathbf{x}-t_i) = \mathbf{w}_{1i}\cdot(\mathbf{x}-t_i)+\mathbf{w}_{1i}\wedge(\mathbf{x}-t_i)$ and $w_{2i}\sigma(\cdot) \Rightarrow \mathbf{w}_{2i}\mathbf{u}(\cdot) = \mathbf{w}_{2i}\cdot\mathbf{u}(\cdot)+\mathbf{w}_{2i}\wedge\mathbf{u}(\cdot)$ then

$$\mathbf{g}(\mathbf{x}) = \sum_{i=1}^{N} \mathbf{w}_{2i}\mathbf{u}[\mathbf{w}_{1i}\cdot(\mathbf{x}-t_i) + \mathbf{w}_{1i}\wedge(\mathbf{x}-t_i)]. \tag{4}$$

In the radial basis function networks, equation (2), the dilation operation (via the diagonal matrix $\mathtt{D_i}$) and the feature enhancement operation can be simultaneously implemented by means of a geometric product, namely $w_{2i}G[\mathtt{D}_i(\mathbf{x} - \mathbf{t}_{1i})] \Rightarrow \boldsymbol{e}_j \boldsymbol{w}_{2_{ji}} G[(\boldsymbol{x} - \boldsymbol{t}_{1i})]$ where i is now for outputs

$$\mathbf{g}_i(\mathbf{x}) = \sum_{j=1}^{N} (\boldsymbol{e}_j \boldsymbol{w}_{2_{ji}}) G[(\boldsymbol{x} - \boldsymbol{t}_{1i})] = \sum_{j=1}^{N} \lambda_j (\boldsymbol{e}_j \boldsymbol{w}_{2_{ji}}) \tag{5}$$

4 Net Architecture and Training Algorithm

The learning procedure for the cases of the previous section has to minimize a metric dependent error function $E(\vec{p})$ where $\vec{p}$ is the vector (not a multivector) which comprises all adjustable parameters. In the first case, the vector $\vec{p}$ (weights and activation values) can be adjusted using the Clifford back propagation training rule [5]. This procedure is unfortunately limited to the Euclidean metric. In contrast the selforganizing Clifford network allows according to the task, if necessary, a different metric. The learning procedure of the net consists basically in the first phase of an unsupervised method for the hidden layer, i.e. a multivector clustering algorithm, and a supervised one for the output layer. The second phase of learning is supervised and if it is still necessary helps to finetune all the net parameters. These phases and the recall mode are explained separately in the next subsections.

4.1 Unsupervised learning

Figure 1 depicts the evolution of the net architecture during selfsorganization. This process is in some aspects similar to the adaptive resonance theory selforganization [7]. At the very beginning the waiting memory and the long term memory have virgin nodes. The input patterns give information of different events and can resonate with a corresponding existing node. This capability of the net is implemented by a resonance detector and control mechanism using a task dependent metric and competitive learning.

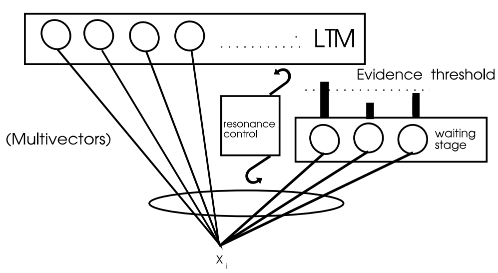

Fig 1.a Dynamic node coding

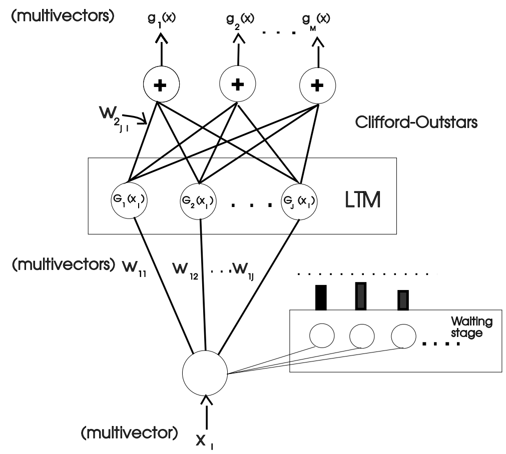

Fig. 1.b Outstars labeling

Note that the geometric algebra approach allows the use of a specific metric for a particular task. When a winner node is selected its weights are smoothly further adjusted. For each new input pattern the control mechanism updates the weights of a resonant node either in the long term memory or in the waiting stage. A candidate node resides in the waiting stage until it surpasses an evidence threshold, then it will be shifted to the long term memory. After the net is stable the parameters of the radial basis functions for each node are computed. These will be used for computing the resonance grade or membership grade (λ_j) of a input multivector with similar nodes. This factor will play an important role in the inhibition effect of the non-resonant nodes (winner-takes-all) during the training of the next layer and during the recall mode.

4.2 Supervised learning

In this stage of the training the multivector weights of the output layer have to be adjusted. Passing again the training patterns, the weights of the outstar of the resonant nodes are adapted using the following simple rule

$$\boldsymbol{w}_{2_{ji}}(k+1) = \boldsymbol{w}_{2_{ji}}(k) + \lambda_j \alpha(k)(\boldsymbol{e}\boldsymbol{o}_i - \boldsymbol{w}_{2_{ji}}(k)) \tag{6}$$

where i is the multivector connection to the output node ith-, λ_j is a constant and indicates the degree of the participation of the node j, $\alpha(k)$ is a gain factor. All multiplications are geometric products and each output $\boldsymbol{o}_i$ supplies a multivector. The multivector $\boldsymbol{e}$ could be set to ${\boldsymbol{w}_{1i}}^{-1}$ or another appropriate multivector as projective split vectors [8] or for the case of a simple multivector association to the scalar unity, i.e. $\boldsymbol{e} = 1$. The projective split can be used to connect the input and output spaces of different dimensions and metric. As a result the invariant properties of the input patterns are enhanced and make possible more observables for the net and in some cases the nonlinearity in one space can be easily transformed to a linear one in the other space representation. Here we can also appreciate the coordinate-free advantages of the geometric algebra. Once an initial solution is found after the first training phase, a supervised learning method can be additionally used in order to finetune all the network parameters. According $E(\vec{p})$ the vector $\vec{p}$, which comprises $\boldsymbol{w}_{1i}$ $\boldsymbol{w}_{2_{ji}}$ will be adjusted after each input and output values $\boldsymbol{x}_k, f(\boldsymbol{x}_k)$ using the metric dependent functional $E(\vec{p}, \boldsymbol{x}_k, f(\boldsymbol{x}_k))$.

4.3 Recall mode

In the recall mode the outputs of the radial basis functions moderate the participation of the resonant nodes at the output energy level. This is captured by a simple equation

$$\boldsymbol{o}_i = \sum_{j=1}^{J} \lambda_j (\boldsymbol{e}_j \boldsymbol{w}_{2_{ji}}) \tag{7}$$

where $\boldsymbol{o}_i$ is the ith output, J is the amount of hidden nodes, λ_j is the degree of the participation of node j and is computed from the radial basis functions. $\boldsymbol{e}_i$ as was mentioned before can be the scalar 1 or a split vector, e.g. $\boldsymbol{w}_{1i}^{-1}$.

5 Experimental Results

The motions of reference frames of joints in robotics can be nicely represented using screws or dual quaternions. The figure 2 depicts the geometric abstraction of the problem. For this experiment the range of movements was limited to a practical narrow area. For the approximation of this mapping a combined structure using two Clifford selforganizing neural networks was implemented. This is presented in figures 3. The two neural networks were set up in the Clifford algebra $\mathcal{C}(V_{0,3})$ [1] accordingly and applied to approximate the mapping between the screw motions of systems A and B. Note that we dedicated two independent networks for each part of the dual quaternion as these dual parts are geometric different. For the selforganizing clustering we used as opposed to the Euclidean metric the second component of the geometric product as a similarity measure. After the selforganization of each network one has recognized a reduced number of long term nodes I and J, i.e clusters of the real and dual part of the dual quaternions. Here we used in fast learning [7] a moderate categorization threshold ρ_i. In order to test the performance of the structure ρ_1 and ρ_2

were varied so that in one case I=J=5 and in the second test I=5 and J=3.

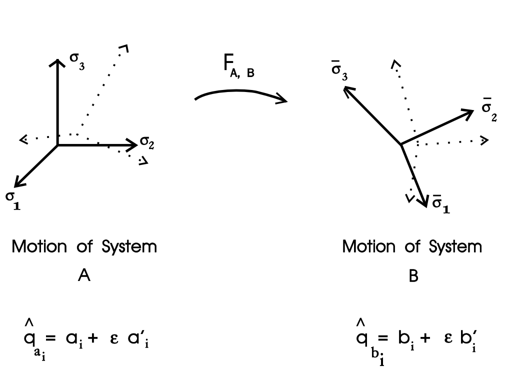

Fig 2. Mapping between two motion spaces

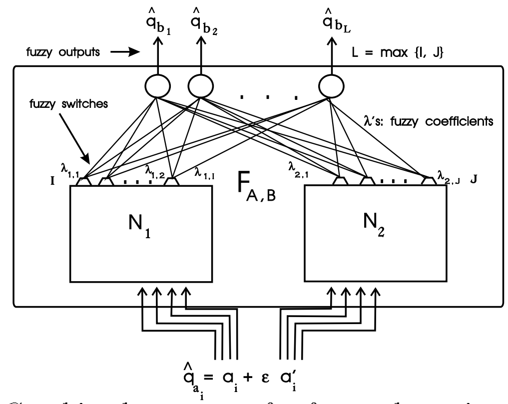

Fig 3. Combined structure for fuzzy clustering

In the supervised phase the radial basis functions were tuned and then the Clifford outstarts were adjusted. The structure is full connected and the weights of the outstars are also quaternions, see figure 3. Note that the amount of the outputs is automatically defined by the bigger number of clusters of the nets, i.e L=MAX(I,J). After the supervised training the net was recalled with previously unseen patterns and due to its nice capability of fuzzy outputs the net was able to follow the deviation of the main classes as expected. Some pattern examples are presented below. The expected dual quaternions at the output are presented in table I.

$Cat.L$	b_0	b_1	b_2	b_3	b'_0	b'_1	b'_2	b_3
1	0.998	0.023	0.031	0.046	-0.061	0.411	0.508	0.754
2	0.927	0.143	0.191	0.287	-0.374	0.528	0.460	0.635
3	0.979	0.076	0.102	0.153	-0.199	0.468	0.494	0.711
4	0.874	0.186	0.248	0.372	-0.484	0.559	0.429	0.572
5	0.771	0.244	0.325	0.488	-0.636	0.589	0.370	0.462

Table I: Expected dual quaternions at the output

The outputs in terms of dual quaternions for each category (Cat.) and its λ_{I-MAX} and λ_{J-MAX} of the hidden layers are presented in the table II for a combined structure with I=J=L=5.

$Cat.$	b_0	b_1	b_2	b_3	b'_0	b'_1	b'_2	b_3	λ_{I-MAX}	λ_{J-MAX}	N_I,N_J
1.1	+0.998	+0.023	+0.031	+0.046	-0.060	+0.411	+0.509	+0.754	0.987	0.793	1,1
1.2	+0.999	+0.018	+0.024	+0.036	-0.047	+0.405	+0.510	+0.758	0.764	0.713	1,1
2.1	+0.927	+0.144	+0.192	+0.288	-0.374	+0.529	+0.460	+0.636	0.531	0.999	2,2
2.2	+0.936	+0.136	+0.181	+0.271	-0.353	+0.522	+0.466	+0.646	0.940	0.697	2,2
3.1	+0.981	+0.075	+0.100	+0.150	-0.195	+0.466	+0.495	+0.714	0.976	0.8850	3,3
3.2	+0.982	+0.073	+0.097	+0.146	-0.190	+0.465	+0.495	+0.715	0.952	0.708	3,3
4.1	+0.877	+0.184	+0.246	+0.369	-0.480	+0.558	+0.431	+0.576	0.988	0.979	4,4
4.2	+0.878	+0.184	+0.245	+0.368	-0.479	+0.558	+0.431	+0.576	0.987	0.975	4,4
5.1	+0.793	+0.234	+0.312	+0.468	-0.609	+0.585	+0.383	+0.485	0.950	0.790	5,5
5.2	+0.795	+0.233	+0.310	+0.466	-0.606	+0.585	+0.384	+0.487	0.945	0.787	5,5

Table II: Dual quaternions at the output structure with I=J=L=5

$Cat.$	b_0	b_1	b_2	b_3	b'_0	b'_1	b'_2	b_3	λ_{I-MAX}	λ_{J-MAX}	N_I,N_J
1-1	+0.999	+0.019	+0.025	+0.037	-0.048	+0.406	+0.510	+0.757	0.987	0.793	1,1
1-2	+0.999	+0.017	+0.022	+0.033	-0.044	+0.404	+0.510	+0.759	0.764	0.713	1,1
2-1	+0.927	+0.144	+0.192	+0.288	-0.374	+0.529	+0.460	+0.636	0.531	0.999	2,3
2-2	+0.964	+0.102	+0.135	+0.203	-0.264	+0.492	+0.484	+0.687	0.940	0.697	2,3
3-1	+0.984	+0.068	+0.091	+0.136	-0.177	+0.460	+0.497	+0.720	1.000	0.952	3,2
3-2	+0.988	+0.060	+0.080	+0.120	-0.156	+0.451	+0.500	+0.727	1.000	0.937	3,2
4-1	+0.880	+0.183	+0.243	+0.365	-0.475	+0.557	+0.432	+0.578	0.997	0.997	4,3
4-2	+0.881	+0.182	+0.243	+0.364	-0.474	+0.557	+0.433	+0.579	0.985	0.995	4,3
5-1	+0.800	+0.231	+0.307	+0.461	-0.600	+0.584	+0.386	+0.491	0.987	0.910	5,2
5-2	+0.801	+0.230	+0.307	+0.460	-0.599	+0.584	+0.387	+0.492	0.980	0.912	5,2

Table III: Dual quaternions at the output structure with I=5, J=3, L=5

The coupled I and J winner nodes (N_I and N_J) for the global assessment at the output are indicated at the right. The results for the structure with I=5, J=3 and L=5 are presented in table

III. Comparing both tables for the three last categories one can see that the combined network with I=J=5 has a better performance than the second with I=5 and J=3. For example, for the 3-2 category the first structure gives the assessments 0.952 for the real part and 0.708 for the dual part of the dual quaternion, whereas the second structure 1.000 and 0.937 respectively. The combined network with I=J=5 is given more information about the approximated membership degree of the dual part because it has dedicated more nodes for clustering. When a bigger ρ_2 is used as in the combined network with I=5 and J=3, the coding of the dual element is more rough affecting the class assessment and eventually the overall performance. Note that there are two nodes which are used for two different classes, i.e. 2,3 and 4,3; and 3,2 and 5,2. It may be possible for other application that a combined structure with I$\neq$J suffices. Therefore it is better that the left and the right modules should use independent ρ's.

6 Conclusion

This paper presents a novel selforganizing type RBF network using the Clifford algebra framework. The authors have shown that the use of geometric algebra helps enormously to improve the potential of network structures and to simplify the learning algorithms. In the network a new type of embedded processing called projective split can be added for feature enhancement and better invariants recognition. This type of neural networks can be cascaded in order to process patterns using different space dimension and metric, the latter being possibly only due the projective split. The potential of such nets working in a specific Clifford algebra $\mathcal{C}(V_{p,q})$ was shown by a simple application of frame coordination in robotics.

References

[1] D. Hestenes and G. Sobczyk. Clifford Algebra to Geometric Calculus: A unified language for mathematics and physics. *D. Reidel*, 1984.

[2] E. Bayro-Corrochano and J. Lasenby. 1995. Object modelling and motion analysis using Clifford algebra. Proceedings of Europe-China Workshop on *Geometric Modeling and Invariants for Computer Vision*, Ed. Roger Mohr and Wu Chengke, Xi'an, China, 143:149, April 1995.

[3] G. Cybenko. Approximation by superposition of a sigmoidal function. *Mathematics of control, signals and systems*, 2, 303:314, 1989.

[4] T. Poggio and F. Girosi. Networks for approximation and learning. *IEEE Proc.*, 78, 9, 1481:1497, Sept. 1990.

[5] J.K. Pearson and D.L. Bisset. Back Propagation in a Clifford Algebra. *Artificial Neural Networks, 2, I. Aleksander and J. Taylor (Ed.)*, 413:416, 1992.

[6] G. M. Georgiou and C. Koutsougeras. Complex domain backpropagation. *IEEE Trans. on Circuits and Systems*, 330:334, 1992.

[7] G.A. Carpenter and S. Grossberg. ART-2: Selforganization of stable category recognition codes for analog input patterns. *Appl. Optics*, 26(23), 4919:4930, 1987.

[8] D. Hestenes. The design of linear algebra and geometry. *Acta Appl. Math.*, 23, 65:93, 1991.