Reduced-Order Shape Optimization Using Offset Surfaces – Supplement

Przemyslaw Musialski* Thomas Auzinger* Michael Birsak* Michael Wimmer* Leif Kobbelt†

*Vienna University of Technology †RWTH Aachen University

Contents

1 Theoretical Background

In this document we provide a brief summary of the theory behind the mass moments, the Divergence Theorem, and the analytic objective function gradient. It is intended to help the reader to more easily follow the paper with consistent notation. Similar considerations, including pseudo-codes, have been also provided by others (e.g., [Marsden and Tromba 2003; Mirtich 1996; Eberly 2010]).

1.1 Mass Properties

Mass Moments. The response of a rigid body to external forces and its dynamic behavior are given by its physical mass moment. Of special interest are the moments of order zero (mass), one (center of mass), and two (moments of inertia). They are further denoted as M_μ and can be expressed as integrals over the volume $\mathcal{V}$ of the given object:

$$M_\mu(\mathcal{V}_\rho) = \int_{\mathcal{V}} \rho(x,y,z)\ \mu(x,y,z)\ \mathrm{d}V\,, \tag{1}$$

where $\mathrm{d}V = \mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z$ is the infinitesimal volumetric unit. The function ρ expresses the mass density, i.e., the mass per a unit volume. The functions μ_n correspond to particular moments at an infinitesimally small unit mass particle and are defined as:

$$\mu_{p,q,r} = x^p y^q z^r\,, \quad 0 \leqslant p,q,r \leqslant 2\,,$$

with $n = p+q+r$ as the *order* of the actual moment. We are interested in the first 10 monomials of order $n \leqslant 2$ that reduce due to symmetry (i.e., $xy = yx$) to the following terms:

$$\boldsymbol{\mu} = \begin{bmatrix} 1 & x & y & z & x^2 & y^2 & z^2 & xy & yz & zx \end{bmatrix}.$$

In the case of a solid rigid body in 3d space, these 10 elementary functions need to be integrated over the entire volume $\mathcal{V}$. For a constant mass density ρ, the mass, which corresponds to the zeroth-order moment, can be computed as

$$M_1 = \rho\|\mathcal{V}\|_{\mathcal{L}^1} = \rho \int_{\mathcal{V}} 1\ \mathrm{d}V\,.$$

Similarly, all 10 interesting moments can be computed respectively by substituting μ with particular monomials from the vector $\boldsymbol{\mu}$, resulting in:

$$\mathbf{M} = \begin{bmatrix} M_1 & M_x & M_y & M_z & M_{x^2} & M_{y^2} & M_{z^2} & M_{xy} & M_{yz} & M_{zx} \end{bmatrix}^\mathsf{T}.$$

The actual properties of an object, in particular mass m, and center of mass $\mathbf{c}$ are finally given as

$$m = M_1\,, \quad \mathbf{c} = \tfrac{1}{M1}\begin{bmatrix} M_x & M_y & M_z \end{bmatrix}^\mathsf{T},$$

and the tensor of inertia as

$$\mathbf{I}' = \begin{bmatrix} M_{y^2}+M_{z^2} & -M_{xy} & -M_{yz} \\ -M_{xy} & M_{x^2}+M_{z^2} & -M_{zx} \\ -M_{yz} & -M_{zx} & M_{y^2}+M_{x^2} \end{bmatrix}.$$

Parallel Axis Theorem. In order to make the tensor of inertia relative to the center of mass of the object, we can apply the parallel axis theorem [Mirtich 1996] as

$$\mathbf{I} = \mathbf{I}' - M_1 \begin{bmatrix} c_y^2+c_z^2 & c_x c_y & c_y c_z \\ c_x c_y & c_x^2+c_z^2 & c_z c_x \\ c_y c_z & c_z c_x & c_y^2+c_x^2 \end{bmatrix} = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{xy} & I_{yy} & I_{zy} \\ I_{xz} & I_{zy} & I_{zz} \end{bmatrix}.$$

Mass Properties. In correspondence with the moments $\mathbf{M}$, we can collect these quantities to arrive at the mass properties

$$\mathbf{P} = \begin{bmatrix} m & c_x & c_y & c_z & I_{xx} & I_{yy} & I_{zz} & I_{xy} & I_{yz} & I_{zx} \end{bmatrix}^\mathsf{T}. \tag{2}$$

Note that the moments $\mathbf{M}$ of an object made out of different parts with moments $\mathbf{M}_i$ can be obtained by their sum, i.e., $\mathbf{M} = \sum_i \mathbf{M}_i$. The corresponding mass properties have to be weighted by their respective masses however.

1.2 Divergence Theorem

An interesting application of our method is efficient computation of the moments M_n of arbitrary freeform surfaces represented as triangular meshes. The idea behind this approach is to reduce the volume integrals as presented in Equation (1) to surface integrals using Gauss's Divergence Theorem [Marsden and Tromba 2003]. The theorem reveals the interesting relation that an integral of the divergence $\nabla\cdot$ of a vector field ϕ over the volume $\mathcal{V}$ is equal to the integral of the unit normal vectors $\mathbf{n}$ over its bounding surface $\partial\mathcal{V}$:

$$\int_{\mathcal{V}} \nabla\cdot\phi\ \mathrm{d}V = \int_{\mathcal{S}} \mathbf{n}\cdot\phi\ \mathrm{d}S\,,$$

where $\mathcal{S} = \partial\mathcal{V}$ is the surface of the object and $\mathrm{d}S$ is the infinitesimal surface area unit. If we assume a constant mass density (i.e. $\rho = 1$ without loss of generality), we can represent the moment monomials μ_n as divergences of suitable vector fields, i.e., $\nabla\cdot\phi_n = \mu_n$. This representation is not unique and we employ the functions as suggested by Mirtich [1996], which have also been used by others [Eberly 2010; Bächer et al. 2014] and combine them to the matrix:

$$\boldsymbol{\phi} = \begin{bmatrix} x & \frac{x^2}{2} & 0 & 0 & \frac{x^3}{3} & 0 & 0 & \frac{x^2 y}{2} & 0 & 0 \\ 0 & 0 & \frac{y^2}{2} & 0 & 0 & \frac{y^3}{3} & 0 & 0 & \frac{y^2 z}{2} & 0 \\ 0 & 0 & 0 & \frac{z^2}{2} & 0 & 0 & \frac{z^3}{3} & 0 & 0 & \frac{z^2 x}{2} \end{bmatrix}.$$

One general assumption we make is that the surface we deal with is a closed and oriented 2-manifold embedded in 3d space $\mathcal{S} =$

$(\mathcal{X}, \mathcal{T})$; in practice represented as a piecewise linear triangle mesh composed of vertices $\mathbf{x} \in \mathcal{X}$ and triangles $\mathcal{T}$, with unit triangle normals $\mathbf{n}_\mathcal{T}$. In such a case, the parameterization and the integration of the surface can be performed for each triangle $\mathcal{T}$ separately

$$\int_\mathcal{S} \mathbf{n} \cdot \phi \, \mathrm{dS} = \sum_{\mathcal{T} \in \mathcal{S}} \int_\mathcal{T} \mathbf{n}_\mathcal{T} \cdot \phi \, \mathrm{dS} \,.$$

For the entire surface $\mathcal{S}$ with mass density ρ, all 10 moments equal to:

$$\mathbf{M}(\mathcal{S}_\rho) = \rho \sum_{\mathcal{T} \in \mathcal{S}} \int_\mathcal{T} \boldsymbol{\phi}^\mathsf{T} \mathbf{n}_\mathcal{T} \, \mathrm{dS} \,. \tag{3}$$

Parameterization. We can define a piecewise parameterization $\mathbf{x}\colon \mathcal{U} \subset \mathbb{R}^2 \to \mathcal{S} \subset \mathbb{R}^3$ for each triangle $\mathcal{T}$ with vertices $\mathbf{p}_0 = (x_0, y_0, z_0)$, $\mathbf{p}_1 = (x_1, y_1, z_1)$, and $\mathbf{p}_2 = (x_2, y_2, z_2)$ as

$$\mathbf{x}(u, v) = \mathbf{p}_0 + u(\mathbf{p}_1 - \mathbf{p}_0) + v(\mathbf{p}_2 - \mathbf{p}_0) \,,$$

with $(u, v) \in \mathcal{U}$. The derivatives w.r.t. $\mathcal{U}$ are

$$\begin{aligned} \mathbf{x}_u &= \partial \mathbf{x} / \partial u = \mathbf{p}_1 - \mathbf{p}_0 \,, \\ \mathbf{x}_v &= \partial \mathbf{x} / \partial v = \mathbf{p}_2 - \mathbf{p}_0 \,, \end{aligned}$$

and the unnormalized normal vector $\tilde{\mathbf{n}}$ is the cross product

$$\tilde{\mathbf{n}} = \mathbf{x}_u \times \mathbf{x}_v \,.$$

With the parameterization $\mathcal{U}$ we can apply the change of variables theorem [Marsden and Tromba 2003] and obtain:

$$\int_\mathcal{S} \mathrm{dS} = \int_\mathcal{U} |\tilde{\mathbf{n}}| \, \mathrm{du} \, \mathrm{dv} \,,$$

where the infinitesimal unit of the area in $\mathcal{U}$ is the normalization factor of the triangle normal. Finally, the particular integrals per triangle reduce to

$$\mathbf{n}_\mathcal{T} \int_\mathcal{T} \phi \, \mathrm{dS} = \tilde{\mathbf{n}}_\mathcal{T} \int_\mathcal{U} \phi \left(\mathbf{x}(u, v) \right) \mathrm{du} \, \mathrm{dv} \,.$$

In practice, we only need to multiply the integral with the coordinate of $\tilde{\mathbf{n}}_\mathcal{T}$ that corresponds to the respective non-zero component of the function ϕ. Due to their simplicity, these integrals can be evaluated analytically, e.g., Eberly [2010] and Bächer et al. [2014] provide pseudo-code for an efficient implementation.

1.3 Derivatives

Through the use of offset surfaces, we are able to utilize continuous optimization in our method to control the mass properties of objects. Furthermore, we can analytically compute the gradient of the objective function without relying on finite difference methodologies. Being significantly more precise this improves both the accuracy and speed of the optimization computations.

Analytic Gradient. In our setting, we have objective functions $f(\mathbf{P})$ taking the mass properties $\mathbf{P}$ as arguments. Through the use the Divergence Theorem and assuming constant mass density for the constituent parts of the object, they are fully determined by the $3n$ coordinates $\mathbf{X}$ of the n mesh vertices $\mathcal{X}$. The vertices, in turn, are shifted along the displacement directions by the displacement values $\boldsymbol{\delta}$. We achieve a significant reduction of the problem's dimensionality by representing the n displacement values with k manifold harmonics modes weighted by the coefficients $\boldsymbol{\alpha}$. In summary, our objective function is given as $f(\boldsymbol{\alpha}) = f(\mathbf{P}(\mathbf{X}(\boldsymbol{\delta}(\boldsymbol{\alpha}))))$.

We can compute an analytic gradient of the objective function f by applying the chain rule:

$$\underbrace{\frac{\partial f}{\partial \boldsymbol{\alpha}}}_{\langle 1 \times k \rangle} = \underbrace{\frac{\partial f}{\partial \mathbf{P}}}_{\langle 1 \times 10 \rangle} \underbrace{\frac{\partial \mathbf{P}}{\partial \mathbf{X}}}_{\langle 10 \times 3n \rangle} \underbrace{\frac{\partial \mathbf{X}}{\partial \boldsymbol{\delta}}}_{\langle 3n \times n \rangle} \underbrace{\frac{\partial \boldsymbol{\delta}}{\partial \boldsymbol{\alpha}}}_{\langle n \times k \rangle} \,,$$

where $\mathbf{X} = \begin{bmatrix} x_1 & y_1 & z_1 & x_2 & y_2 & z_2 & \dots & x_n & y_n & z_n \end{bmatrix}^\mathsf{T}$ are the n vertices of the original surface concatenated into a $\langle 3n \times 1 \rangle$ vector.

The partial derivatives $\partial \mathbf{P} / \partial \mathbf{X}$ of the ten properties $\mathbf{P}$ w.r.t. surface vertices $\mathbf{X}$ result in a $\langle 10 \times 3n \rangle$ matrix. Note that $\mathbf{P}$'s are functions of the integrals M_μ from Eq. (1). The derivatives of $\partial \mathbf{X} / \partial \boldsymbol{\delta}$ are a sparse matrix of the size $\langle 3n \times n \rangle$ that contains in each column one displacement vector $\boldsymbol{\nu}$, i.e, for $\underline{\boldsymbol{\nu}}$:

$$\frac{\partial \mathbf{X}}{\partial \underline{\boldsymbol{\delta}}} = \begin{bmatrix} \underline{\nu}_{x_1} & \underline{\nu}_{y_1} & \underline{\nu}_{z_1} & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \underline{\nu}_{x_2} & \underline{\nu}_{y_2} & \underline{\nu}_{z_2} & \cdots & 0 & 0 & 0 \\ & \vdots & & & & & \ddots & & \vdots & \\ 0 & 0 & 0 & 0 & 0 & 0 & \cdots & \underline{\nu}_{x_n} & \underline{\nu}_{y_n} & \underline{\nu}_{z_n} \end{bmatrix}^\mathsf{T} .$$

Finally, the derivatives $\partial \boldsymbol{\delta} / \partial \boldsymbol{\alpha}$ w.r.t. the coefficients $\boldsymbol{\alpha}$ are the first k manifold harmonics basis vectors $\boldsymbol{\Gamma}_k$ themselves.

Double-Sided Offset. The sizes given above account for only one-sided offset surface. However, double-sided offset can be performed in one step by extending the matrices appropriately.

The differences start at the term $\partial \mathbf{P} / \partial \mathbf{X}$, where the derivatives need to be computed with respect to vertex positions. If the starting point of displacement in both directions is the same surface both the inner $\underline{\mathbf{X}}$ and outer vertices $\overline{\mathbf{X}}$ are in practice the same (both just $\mathbf{X}$)[1]. If two distinct surfaces are taken as initial values, their vertices need to be taken into account:

$$\partial \mathbf{P} / \partial \mathbf{X} = \begin{bmatrix} \partial \mathbf{P} / \partial \underline{\mathbf{X}} & \partial \mathbf{P} / \partial \overline{\mathbf{X}} \end{bmatrix}_{\langle 10 \times 6n \rangle} \,,$$

$$\frac{\partial \mathbf{X}}{\partial \boldsymbol{\delta}} = \begin{bmatrix} \partial \underline{\mathbf{X}} / \partial \underline{\boldsymbol{\delta}} & 0 \\ 0 & \partial \overline{\mathbf{X}} / \partial \overline{\boldsymbol{\delta}} \end{bmatrix}_{\langle 6n \times 2n \rangle} \,,$$

and

$$\frac{\partial \boldsymbol{\delta}}{\partial \boldsymbol{\alpha}} = \begin{bmatrix} \boldsymbol{\Gamma}_k & 0 \\ 0 & \boldsymbol{\Gamma}_k \end{bmatrix}_{\langle 2n \times 2k \rangle} .$$

The composite partial derivatives $\partial \mathbf{P} / \partial \boldsymbol{\alpha}$ of all mass properties $\mathbf{P}$ w.r.t. $\boldsymbol{\alpha}$ thus reduce to a $\langle 10 \times 2k \rangle$ sized matrix.

2 Fabrication Comparison

The smooth surfaces of the inner voids that our method generates have a considerable impact on the fabrication process when using FMD 3d printing. In comparison to the voxelized interiors of previous works we need roughly half the support material and half the print time (see Table 1). This can be explained by the fact that the sharp voxel boundaries produce many horizontal overhangs that need to be supported throughout the model. Both the speed of the print head and its traveled distance generally larger when drawing voxel boundaries in comparison with smooth curves.

3 Optimization Parameters

The settings and quantitative results of our examples are given in Table 2.

[1] For better readability we drop the upper/lower lines, such that $\mathbf{X} = \overline{\mathbf{X}}$.

Models of Figure 6	Model material [cm^3]	Support material [cm^3]	Duration [hh:mm]
Ours	26.98	8.28	2:24
[Prévost et al. 2013]	15.55	8.25	2:23
Platform ours		4.00	0:14
Platform [Prévost et al. 2013]		3.04	0:10
Ours without Platform	26.98	4.28	2:10
[Prévost et al. 2013] without Platform	15.55	5.21	2:13
		Relative material [%]	Relative duration [hh:mm cm^{-3}]
Ours without Platform		**16.7**	**0:05**
[Prévost et al. 2013] without Platform		**35.7**	**0:09**

Table 1: *Comparison of the print requirements for the models shown in Figure 6 of the paper. We give the material requirements and print duration on a* Dimension uPrint Plus *FDM printer where we subtract the build process of a platform out of support material, which is required by the printer. Since the models exhibit significantly different sizes due to the respective deformation models used by the optimization, we normalize both support material and duration by the volume of the actual object. We observe that in comparison to Prévost et al. [2013], we need only half the support material and printing time.*

References

BÄCHER, M., WHITING, E., BICKEL, B., AND SORKINE-HORNUNG, O. 2014. Spin-It: Optimizing Moment of Inertia for Spinnable Objects. *ACM Transactions on Graphics 33*, 4 (July), 1–10.

EBERLY, D. H. 2010. *Game physics*, 2. edition ed. Morgan Kaufmann, Burlington, Mass.

MARSDEN, J. E., AND TROMBA, A. 2003. *Vector Calculus*. W. H. Freeman.

MIRTICH, B. 1996. Fast and Accurate Computation of Polyhedral Mass Properties. *Journal of Graphics Tools 1*, 2 (Jan.), 31–50.

PRÉVOST, R., WHITING, E., LEFEBVRE, S., AND SORKINE-HORNUNG, O. 2013. Make It Stand: Balancing Shapes for 3D Fabrication. *ACM Transactions on Graphics 32*, 4 (July), 1.

Figure	f_{min}	m	c_x	c_y	c_z	I_{x^2}	I_{y^2}	I_{z^2}	I_{xy}	I_{yz}	I_{zx}
1.1	8.8464	1.0272	3.0363	36.3237	0	13.7833	-13.7636	42.7504	0.0002	2.6353	0.0001
1.2	2.947	11.5255	0.7236	0	2.4233	40.6745	47.4858	15.6111	-14.9261	0	0.0001
1.3	5.9109	22.1134	0.5548	-0.0003	2.3156	61.4414	69.9063	25.6067	-20.5774	-0.0014	-0.0045
6.1	5.3066	109.7995	-0.05	0.05	5.3016	1174.715	1158.521	1144.931	191.5154	186.5148	14.9104
6.2	3.8319	11.7715	-1	-0.0009	2.8319	66.5334	106.5635	44.8086	38.2834	-0.0891	0.2135
6.3	10.1585	16.3968	0.0002	0	6.3376	143.4909	203.9611	87.0337	55.7939	-0.5091	13.3837
7	0.8223	2.2567	0	0	0.8223	1.46	1.4411	1.0268	-0.0003	0.0014	0.0066
8	0.9129	0.4259	0	0	0.3799	0.207	0.1987	0.3504	0	0.0168	0
9	-0.0505	2.1874	-0.1003	0.0006	-0.0567	0.5125	1.4401	1.2397	0.0374	0.008	0.0022
12	2.5188	21.777	0.6262	0.0013	2.1266	65.7607	100.038	66.7332	-33.5538	-0.1162	0.0289
13	11.4486	1.0741	-0.0002	-0.0001	1.1313	0.3541	0.4104	0.5206	0	0	0
14.1	3.7702	6.7607	0.4423	-0.1484	3.5526	45.2525	47.2934	9.7826	-2.296	0.2528	-6.7172
14.2	6.1278	15.1399	0.1	0.0273	5.1857	112.2568	117.6003	28.6029	5.3453	1.8889	-16.5581

Figure	vertices	iter	t_{solve} [s]	t_{prep} [s]	objective	offsets	k
1.1	12160	0	9.2191	0.0409	-	inner	-
1.2	12160	7	2.9204	0.3766	(5)	inner	4
1.3	12160	11	11.2086	0.435	(6)	both	10
6.1	6161	35	16.3654	0.6336	(5)	inner	42
6.2	4663	11	2.5614	0.4407	(5)	inner	32
6.3	85793	22	227.3832	3.2361	(5)	both	36
7	2316	25	4.0786	0.0975	(7)	inner	36
8	20438	14	54.4084	1.1668	(8)	both	48
9	8926	27	101.2219	0.296	(9)	both	36
12	6022	30	4.8376	0.1741	(5)	inner	26
13	9178	15	3.4449	0.2525	(8)	inner	24
14.1	43183	45	12.4185	1.8768	(5)	inner	96
14.2	43183	39	681.4396	1.8283	(5)	both	96

Table 2: *Settings and quantitative results of our example. The number of iterations of our non-linear solver are given together with the solution, the solution time (t_{solve}), and the preprocessing time (t_{solve}, includes Laplcaian and eigs computation, not the skeleton), the number k of utilized manifold harmonics modes.*

exitflag	k	iter	time	f_{min}	m	c_x	c_y	c_z	I_{x^2}	I_{y^2}	I_{z^2}	I_{xy}	I_{yz}	I_{zx}
5	-	7	86.6995	11.2798	0.768	0	0	0.8168	0.1316	0.1792	0.2154	0.0001	0	0
-2	10	2	0.0518	16.3126	0.4468	0.0016	0	0.8237	0.1068	0.1393	0.139	0.0084	0	0
-2	14	4	0.1146	16.2742	0.4671	-0.0006	0	0.821	0.1128	0.1395	0.1423	0.0054	0	0
-2	18	4	0.1467	15.7559	0.5088	0.0008	0	0.8226	0.1181	0.1506	0.1546	0.0047	-0.0002	0
-2	22	4	0.1889	15.5082	0.5054	0.0004	0	0.8176	0.1169	0.1494	0.1544	0.0048	-0.0004	0
-2	26	9	0.5487	16.4028	0.5514	0.0002	0	0.8187	0.1091	0.1641	0.156	0.0009	0	0
-2	30	10	0.7054	16.2094	0.5454	-0.0005	0	0.8225	0.1077	0.1632	0.1557	0.001	0	0
5	34	7	0.4368	14.5884	0.6211	0	0	0.8153	0.1267	0.1602	0.1721	0	0	0
5	38	11	0.5901	13.9134	0.6413	0	0	0.814	0.1262	0.1653	0.1797	0	0	0
5	42	22	1.4623	13.5241	0.6457	0	0	0.8173	0.1272	0.1667	0.1839	0	0.0002	0
5	46	9	0.9293	13.4684	0.6426	0	0	0.8184	0.1264	0.1666	0.1838	0	0	0
5	50	19	2.2558	13.3824	0.6237	0	0	0.8207	0.1214	0.1663	0.1815	0	0.0002	0
5	54	15	1.4985	13.3653	0.6413	0	0	0.8206	0.1245	0.1677	0.1843	0	0	0
5	58	8	0.7569	13.089	0.6324	0	0	0.8168	0.1226	0.167	0.1848	0	0.0001	0
5	62	7	0.9368	12.8216	0.6319	0	0	0.8188	0.1205	0.1688	0.187	0	0	0
5	66	4	1.1524	12.8103	0.6357	0.0006	0	0.8169	0.1213	0.1692	0.1878	0	0	0
5	70	9	1.6226	12.7835	0.6376	0	0	0.8234	0.1233	0.1676	0.1879	0	0	0
5	74	7	2.0703	12.626	0.675	0	0	0.824	0.1279	0.172	0.1951	0	0.0001	0
5	78	10	2.9215	12.2553	0.6783	0	0	0.8173	0.1284	0.1719	0.1984	0	0.0002	0
5	82	9	3.0327	12.1865	0.6926	0.0001	0	0.8209	0.1284	0.1738	0.2005	0	-0.0002	0
5	86	9	2.8716	12.0956	0.7084	0	0	0.8163	0.1325	0.1737	0.2036	0	0.0002	0
5	90	9	3.2759	12.018	0.7184	0.0001	0	0.8167	0.1336	0.1733	0.2046	0	-0.0001	0
5	94	8	4.389	12.0315	0.7161	0	0	0.8168	0.1331	0.1737	0.2045	0	-0.0001	0
5	98	8	4.2123	12.0314	0.7171	0	0	0.8158	0.1333	0.1737	0.2047	0	-0.0001	0
5	750	18	92.9968	11.2725	0.7691	0	0	0.8165	0.1308	0.1781	0.2141	0	0	0

Table 3: *Numbers of the test presented in Figures 10 and 11 in the paper with a small top with $n = 750$ vertices. First row shows the results of full optimization without the usage of order-reductions, and the last row shows the results of a full-basis optimization (k=n).*