Computer Aided Geometric Design 42 (2016) 1–6

Contents lists available at ScienceDirect

Computer Aided Geometric Design

www.elsevier.com/locate/cagd

Short communication

Pyramid algorithms for barycentric rational interpolation ☆

Kai Hormann *, Scott Schaefer

ARTICLE INFO

Article history:
Received 3 June 2015
Received in revised form 2 December 2015
Accepted 10 December 2015
Available online 4 January 2016

Keywords:
Barycentric rational interpolation
Neville's algorithm

ABSTRACT

We present a new perspective on the Floater–Hormann interpolant. This interpolant is rational of degree (n, d), reproduces polynomials of degree d, and has no real poles. By casting the evaluation of this interpolant as a pyramid algorithm, we first demonstrate a close relation to Neville's algorithm. We then derive an $O(nd)$ algorithm for computing the barycentric weights of the Floater–Hormann interpolant, which improves upon the original $O(nd^2)$ construction.

© 2015 Elsevier B.V. All rights reserved.

1. Introduction

Given the $n+1$ interpolation nodes $x_0 < x_1 < \cdots < x_n$ and the associated data $f_0, f_1, \ldots, f_n$, there are two ways to write the rational Floater–Hormann interpolant (Floater and Hormann, 2007) of degree $d \le n$. On the one hand, it can be expressed as the blend

$$r(x) = \frac{\sum_{i=0}^{n-d} \lambda_i(x) p_i(x)}{\sum_{i=0}^{n-d} \lambda_i(x)} \tag{1}$$

of the polynomials p_i of degree d, which locally interpolate the data $f_i, \ldots, f_{i+d}$, with weighting functions

$$\lambda_i(x) = \frac{(-1)^i}{(x-x_i)\cdots(x-x_{i+d})}.$$

On the other hand, it can be written in the barycentric form

$$r(x) = \sum_{i=0}^{n} \frac{(-1)^i}{x-x_i} w_i f_i \Bigg/ \sum_{i=0}^{n} \frac{(-1)^i}{x-x_i} w_i \tag{2}$$

with positive weights

$$w_i = \sum_{j=\max(0,i-d)}^{\min(i,n-d)} \prod_{k=j,k\neq i}^{j+d} \frac{1}{|x_i - x_k|}. \tag{3}$$

The barycentric form is particularly suited for evaluating the interpolant r in $O(n)$ time, once the weights w_i, which depend only on the nodes x_i and not on the data f_i, have been precomputed. Using (3), these weights can be determined in $O(nd^2)$

☆ This paper has been recommended for acceptance by Falai Chen.

* Corresponding author. Tel.: +41 58 666 43 27; fax: +41 58 666 45 36.
E-mail address: kai.hormann@usi.ch (K. Hormann).

http://dx.doi.org/10.1016/j.cagd.2015.12.004
0167-8396/© 2015 Elsevier B.V. All rights reserved.

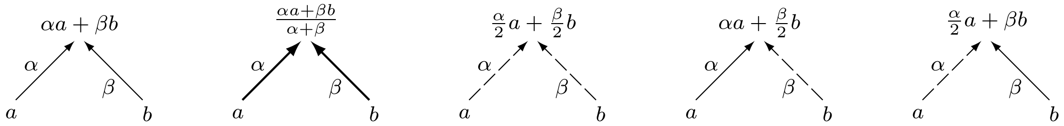

Fig. 1. Pyramid notation for linear combinations. Thick arrows indicate affine combinations, where we omit the normalization factors of the weights to keep the diagram less cluttered. Dashed arrows indicate that the weights need to be multiplied with 1/2.

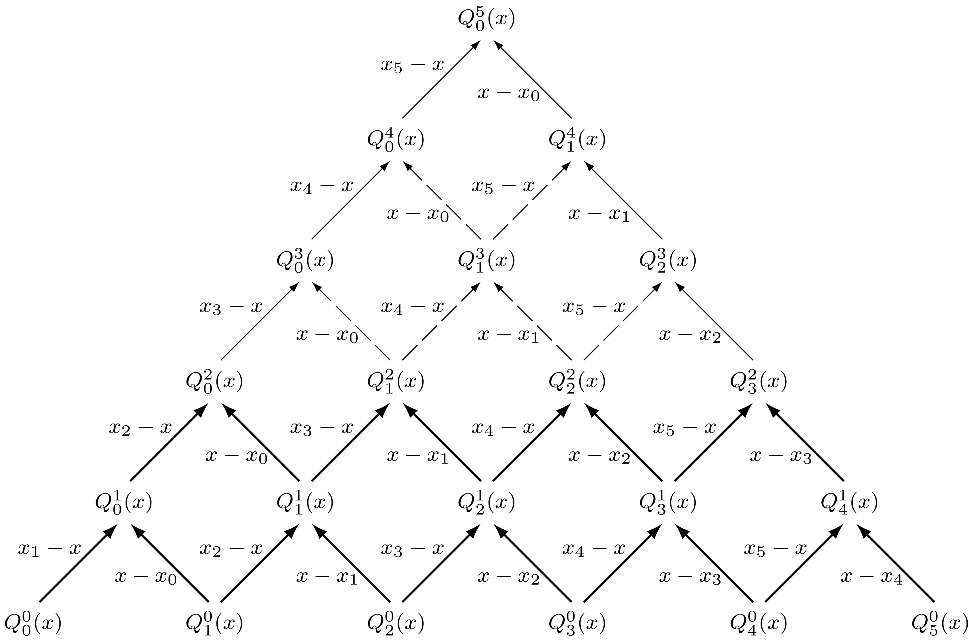

Fig. 2. Example of the pyramid algorithm for Floater–Hormann interpolation with $n = 5$ and $d = 2$.

steps, which is exactly how common libraries like *Numerical Recipes*[1] (Teukolsky et al., 2007) and *ALGLIB* (Bochkanov, 2015) perform the computation.

We present two novel procedures for evaluating $r(x)$, which are inspired by Ron Goldman's pyramid algorithms (Goldman, 2003) for the evaluation of polynomial and spline curves. While the first is tailored for Floater–Hormann interpolants and exploits the representation of r in (1), the second is based on (2) and works for general barycentric rational interpolants of degree (n, n) with arbitrary weights w_i. Both algorithms require $O(n^2)$ operations and are closely related to Neville's algorithm for constructing interpolating polynomials of degree n. They further lead to a novel $O(nd)$ algorithm for computing the weights in (3).

2. Evaluating the rational interpolant

Using the notation in Fig. 1, the Floater–Hormann interpolant in (1) can be evaluated by the pyramid algorithm in Fig. 2, which is a slight variation of Neville's algorithm (Goldman, 2003, Chapter 2.2), where the weights in the top $n - d$ rows of the pyramid are not normalized and the weights at the interior edges in these rows are multiplied with an additional factor of 1/2. That is, for some given evaluation parameter x, we start with the initial data

$$Q_i^0(x) = (f_i, 1), \qquad i = 0, \ldots, n$$

and compute the bottom d rows of the pyramid with Neville's algorithm as

$$Q_i^\ell(x) = \frac{x_{i+\ell} - x}{x_{i+\ell} - x_i} Q_i^{\ell-1}(x) + \frac{x - x_i}{x_{i+\ell} - x_i} Q_{i+1}^{\ell-1}(x), \qquad i = 0, \ldots, n - \ell,$$

for $\ell = 1, \ldots, d$, resulting in the values

$$Q_i^d(x) = (p_i(x), 1), \qquad i = 0, \ldots, n - d.$$

[1] The claim in Teukolsky et al. (2007), page 128, that "the workload to construct the weights is of order $O(nd)$ operations" is wrong, because the given code just implements the formula in (3) in a straightforward way and is clearly of order $O(nd^2)$.

Download English Version:

https://daneshyari.com/en/article/441404

Download Persian Version:

https://daneshyari.com/article/441404

Daneshyari.com