American Journal of Operations Research, 2016, 6, 468-479
http://www.scirp.org/journal/ajor
ISSN Online: 2160-8849
ISSN Print: 2160-8830

Dual Based Procedures for Un-Capacitated Minimum Cost Flow Problem

Priyank Sinha, Renduchintala Raghavendra Kumar Sharma

Industrial and Management Engineering Department, IIT Kanpur, Kanpur, India
Email: spriyank@iitk.ac.in, rrks@iitk.ac.in

How to cite this paper: Sinha, P. and Sharma, R.R.K. (2016) Dual Based Procedures for Un-Capacitated Minimum Cost Flow Problem. *American Journal of Operations Research*, **6**, 468-479.
http://dx.doi.org/10.4236/ajor.2016.66043

Received: October 20, 2016
Accepted: November 15, 2016
Published: November 18, 2016

Abstract

In this article, we devise two dual based methods for obtaining very good solution to a single stage un-capacitated minimum cost flow problem. These methods are an improvement to the methods already developed by Sharma and Saxena [1]. We further develop a method to extract a very good primal solution from a given dual solution. We later demonstrate the efficacies and the significance of these methods on 150 random problems.

Keywords

Min Cost Flow, Transshipment, Dual, Primal

1. Introduction

Un-capacitated min cost flow problem is a special case of min cost flow problem in which arc capacities are assumed to be infinite. Weintraub [2] developed a variant of negative cycle algorithm which searched for the most negative cycle and subsequently introduced it into the feasible flow at each iteration. Later a strongly polynomial time algorithm for min cost flow was developed by Tardos [3] with a computational complexity of $O(m^4)$. Enhanced capacity scaling algorithm can be used to solve Transshipment problem with computational complexity of O(n log (n) S(n,m)) (Ahuja *et al.* [4]). Tardos [3] developed cost scaling algorithm with the computational complexity of $O(n^3 \log n)$. In this algorithm, dual optimality conditions are relaxed to form e-optimality conditions. Thus the best primal based methods solve un-capacitated min cost flow problem in $O(n^3 \log (n))$. Recently Juman [5] has presented a heuristic with $O(n^3)$ running time to solve un-capacitated transportation problem, and is shown to perform better than VAM.

Successive shortest path algorithm was developed by Busakar and Gowan [6]. This algorithm maintains dual feasibility at each step and iteratively achieves primal feasibility. Edmonds and Karp [7] proposed the first polynomial time algorithm by modifying the method to calculate shortest paths, to solve min cost flow problem with computational complexity of O((n + m) log U). Dual simplex for network flow was first analyzed by Hegason and Kennington [8]. Plotkin and Tardos [3] improved the pivoting strategy with (m^2 log n) bound over the pivoting strategy proposed by Orlin [9]. This improves the number of pivot steps required in dual simplex algorithm. This algorithm runs in O(m^3 log(n)) time. Ali *et al.* [10] have demonstrated that an efficient execution of each pivot in dual based algorithm requires less iterations as compared to primal based algorithms. This holds true even for the re-optimization process. However, computational effort required per pivot may be higher. Sharma and Sharma [11] have given a new dual based procedure that has obtained solutions within 85% of the optimal.

Sharma and Saxena [1] have posed the transshipment problem differently. We use the formulation proposed by Sharma and Saxena [1]. We then modify the dual based methods developed by them to obtain better solutions with the same complexity of O(n^2) and O(n^3) respectively. We further devise a method to obtain a good primal solution from the dual solutions already obtained. Empirical results on the random 150 problems are given in **Appendix 1**.

2. Problem Formulation

We next present the mathematical formulation of the primal problem and dual problem respectively.

2.1. Constants of Problem

D_k refers to the demand at the K^{th} demand node, while d_k is the demand at market k as a fraction of total market demand. Hence we have $d_k = D_k \Big/ \left[\sum_{k=1}^{K} D_k\right]$ and $\sum_{k=1}^{K} d_k = 1$, where K is the total number of demand nodes. Similarly S_i refers to units available for transportation at the source node i and $s_i = S_i \Big/ \sum_{k=1}^{K} D_k$. If the problem is balanced, then we have $\sum_{i=1}^{I} s_i = \sum_{k=1}^{K} d_k$, I is the total number of supply nodes. J is total number of transshipment nodes. $C1_{ij}$ and $C2_{jk}$ is the cost of transporting $\sum_{k=1}^{K} D_k$ units from node to j and j to k respectively.

2.2. Decision Variables

$x1_{ij}$ and $x2_{jk}$ is the number of units transported from node i to node j and j to k respectively. We also have $X1_{ij} = x1_{ij} \Big/ \left[\sum_{k=1}^{K} D_k\right]$ and $X2_{jk} = x2_{jk} \Big/ \left[\sum_{k=1}^{K} D_k\right]$.

2.3. Primal (P)

$$\text{Minimize} \sum_{i=1}^{I}\sum_{j=1}^{J} X1_{ij}C1_{ij} + \sum_{j=1}^{J}\sum_{k=1}^{K} X2_{jk}C2_{jk}$$

$$\text{Subject to}: \sum_{i=1}^{I}\sum_{j=1}^{J} X_{ij} = 1 \ \ \forall i, j \tag{1}$$

$$\sum_{j=1}^{J}\sum_{k=1}^{K} X2_{jk} = 1 \ \ \forall j, k. \tag{2}$$

$$\sum_{j=1}^{J} X1_{ij} \ge -s_i \ \ \forall i \tag{3}$$

$$\sum_{j=1}^{J} X2_{jk} \ge -d_k \ \ \forall k \tag{4}$$

$$\sum_{k=1}^{K} X2_{jk} - \sum_{i=1}^{I} X1_{ij} = 0 \ \ \forall j \tag{5}$$

$$X1_{ij}, X2_{jk} \ge 0 \ \ \forall i, j, k.$$

In this formulation we assume flows only in the forward direction. Equation (1) ensures that entire supply is transported to meet the demand, which is valid for the balanced problem. Equation (2) ensures that the total demand is met by the supply. Equations (3) and (4) ensure that individual supply and demand constraints are satisfied, while Equation (4) ensures that no inventory is built at any transshipment node.

2.4. Dual of the Problem (DP)

In this section we present the dual of the problem P. We associate $V1, V2, U_i, V_k, W_j$ as the dual variables corresponding to (1), (2), (3), (4), (5) respectively. We first state the dual of the problem as DP and then divide it into two parts as DP-source and DP-sink for computational simplicity.

DP

$$\text{Maximize}: V1 + V2 - \sum_{i=1}^{I} s_i U_i - \sum_{k=1}^{K} d_k V_k$$

$$\text{Subject to}: V1 - U_i - W_j \le C1_{ij} \ \ \forall i, j \tag{6}$$

$$V2 - V_k + W_j \le C2_{jk} \ \ \forall j, k \tag{7}$$

$U_i, V_k \ge 0$, $V1, V2, W_j$ Unrestricted in sign

DP-source

$$\text{Maximize}: V1 - \sum_{i=1}^{I} s_i U_i$$

$$\text{Subject to}: V1 - U_i - W_j \le C1_{ij} \ \ \forall i, j \tag{8}$$

$U_i \ge 0$, $V1$ and W_j unrestricted in sign.

DP-sink

$$\text{Maximize}: V2 - \sum_{k=1}^{K} d_k V_k$$

$$\text{Subject to}: V2 - V_k + W_j \le C2_{jk} \ \ \forall j, k \tag{9}$$

$V_k \ge 0$, $V2$ and W_j unrestricted in sign.

3. Few Theoretical Results

We start with development of the heuristic for the dual solution, and then move on to develop the heuristic for the primal. Computational attractiveness of these results will be demonstrated in the later sections through empirical testing. Well known dual based approaches (Orlin [9], Plotkin and Tardos [3] and Ali *et al.* [10]) can be used for our solution to get an advanced start while solving the transshipment problem. We begin by defining the set SPS which is as under-

SPS = {SP_{ik}:SP_{ik} is the shortest path between i and k}.

Problem (TP)

$$\text{Minimize}: \sum_{i=1}^{I}\sum_{k=1}^{K} X_{ik} SP_{ik}$$

$$\text{Subject to}: \sum_{k=1}^{K} X_{ik} = s_i \quad \forall i \tag{10}$$

$$\sum_{i=1}^{I} X_{ik} = d_k \quad \forall k \tag{11}$$

and $X_{ij} \geq 0$, $\forall i, k$.

Theorem 1: Optimal solution of problem TP is equal to optimal solution to problem P.

Proof: Since upper value of the flow is unbounded, hence optimal flow for a pair of source node and sink node will be on SP_{ik}. This ensures that any further reduction in the objective value is not possible. Therefore problem TP gives the optimal solution to problem P. **Hence proved**.

4. Solution Procedure

4.1. Heuristic to Solve Dual of the Problem (H1)

DP-source and DP-sink are equivalent in structure to DRP1 in Sharma and Murlidhar [12]. Sharma and Murlidhar [12] have given an efficient algorithm to solve DRP1 which can be modified to solve DP-source and DP-sink.

Step 1. DP-source and DP-sink can be rewritten as under

DP-source

$$\text{Maximize}: V1 - \sum_{i=1}^{I} s_i U_i$$

$$\text{Subject to}: V1 - U_i \leq C1_{ij} + W_j \tag{12}$$

$U_i \geq 0$, $V1$ and W_j unrestricted in sign.

DP-sink

$$\text{Maximize}: V2 - \sum_{k=1}^{K} d_k V_k$$

$$\text{Subject to}: V2 - V_k \leq C2_{jk} - W_j \tag{13}$$

$V_k \geq 0$, $V2$ and W_j unrestricted in sign.

Step 2. Find $d_{k1}^* = \min_j \left(C1_{ij} + W_j \right)$ and $d_{k2}^* = \min_j \left(C2_{jk} - W_j \right)$ $\forall$ all *i*, *j*, *k* and $W_j = 0$ and remove all the redundant constraints in DP-source and DP-sink (Equations (8) and (9)). In case of tie, only one equation is retained while others are eliminated. This reduces the DP-source and DP-sink to the following form:

DP-source

Maximize: $V1 - \sum_{i=1}^{I} s_i U_i$

s.t $V1 - U_i \le d_{k1}^*$ $\forall i$

DP-sink

Maximize: $V2 - \sum_{k=1}^{K} d_k V_k$ $\forall k$

s.t. $V2 - V_k \le d_{k2}^*$

d_{k1}^* and d_{k2}^* represent the least cost transportation route between source and transshipment node and transshipment node and sink respectively.

Step 3. We sort the values of d_k^* in an increasing order and re-index such that $\left(d_{k1}^* + d_{k2}^* \right)_j \le d_{r+1}^*$ $\forall r = 1, \cdots, \max(i,k)$.

Step 4. Since $\sum_{k=1}^{K} d_k = 1$ and $\sum_{i=1}^{I} s_i = 1$, we let $V1 = d_{k1}^*$, $U_i = V1 - d_{k1}^*$, $V2 = d_{k2}^*$ and $V_k = V2 - d_{k2}^*$. Solution to the problem is given by

$$\sum_{i=1}^{I} \sum_{k=1}^{K} d_{k1}^* \left(\min_j \left(C1_{ij} + W_j \right) \right) + d_{k2}^* \left(\min_j \left(C2_{jk} - W_j \right) \right).$$

We repeat the whole procedure for different increases in values of W_j $\forall$ all *j* and retain the best solution.

It may be noted that when we increase/decrease the value of W_j $\forall$ *j*, DP-source increases while DP-sink decreases as per the structure of DP-source and DP-sink. Actually all four possibilities are there for a general case. Our algorithm here intends to balance value of W_j for the best trade-off possible.

Result 1: Computational complexity of A1 is O(n^2).

Proof: Complexity of algorithm is dominated by step 2 which can be solved in O(n2) time.

4.2. Heuristic to Solve Dual of the Problem (H2)

In the previous algorithm, we tinkered with value of W_j along SP_{ik}. There is no reason as to why we should not tinker with the values of U_i, V_k, W_j along SP_{ik}. $\text{Min}\left(C1_{ij} - V1 + U_i + W_j \right)$ and $\text{Min}\left(C2_{jk} - V2 + V_k - W_j \right)$ are achieved simultaneously along SP_{ik} in this method. $\left(C1_{ij} - V1 + U_i + W_j \right)$ and $\left(C2_{jk} - V2 + V_k - W_j \right)$ are defined as source slack and sink slack respectively $\left(S_{so}, S_{si} \right)$ in the later sections. Next we describe this heuristic in detail.

Step 0: Set $W_j = 0$ $\forall$ *j* = 1, …, *J*.

Step 1: Compute max_value of DP_source and DP_sink, set current_best_DP = objective function value of (DP source + DP-sink), set *j* = 0.

Step 2: *j* = *j* + 1; if *j* > *J* then stop or else go to step 3.

Step 3: Increase value of W_j in steps and compute for each value of W_j: max_value of DP-source and DP-sink.

Step 4: Set current_DP = objective function value of (DP-source + DP-sink).

If current_DP > current_best_DP then current_best_DP = current_DP go to step 3, else go to step 2.

Result 2: Heuristic 2 runs in O(n^3) time.

Proof: Complexity of the step is heuristic is dominated by step 3 which can be completed in O(n^3) steps.

4.3. Development of the Heuristic to Obtain a Good Primal Solution (H3)

In this section we will develop a primal heuristic by utilizing the complimentary slackness condition. This heuristic extracts a good primal solution from a good dual solution by utilizing complimentary slackness condition. Let us denote the solution of DP by V_{sso}, V_{ssi}, $U = \{U_i, \forall i = 1,\cdots, I\}$, $V = \{V_k, \forall k = 1,\cdots, K\}$. We further define l_j, $j = 1,\cdots, J$ and l_k, $k = 1,\cdots, k$ as $l_j = \{ j : Csot_{ij} + W_j = d^*_{k1} \}$ and $l_k = \{ j : Ctsi_{jk} - W_j = d^*_{k2} \}$ where $d^*_{k1} = \min_j (Csot_{ij} + W_j)$ and $d^*_{k2} = \min_j (Ctsi_{jk} - W_j)$. Slack S_{so} and S_{si} is defined as following:

$S_{so} = |Csot_{ij} - V_{sso} + U_i + W_j|$ and $S_{si} = |Ctsi_{jk} - V_{ssi} + V_k - W_j|$. If $S_{so} = 0$ and $S_{si} = 0$, then $X_{ij} \geq 0$ and $X_{jk} \geq 0$. X_{ij} and X_{jk} can assume a positive value if for the corresponding i and k, we have $i \in l_j$ and $k \in l_k$. According to the complimentary slackness condition, $X_{ij} = 0$ and $X_{jk} = 0$ when $i \neq l_j$ and $k \neq l_k$. Let S_{so} and S_{si} be the source and sink slacks respectively, $SP_{ik} = \min_j \left(\sum_{i,j} S_{so} + \sum_{j,k} S_{si} \right)$ and $DN_{ik} = SP_{ik} \times X_{ik}$. DN_{ik} is then referred to as deviation number. If $SP_{ik} = 0$, then we can send a positive flow along this arc without violating the Complimentary slackness property. However if $SP_{ik} > 0$, then flow along this has to be zero if complimentary slackness property is not to be violated. As we are working with good dual solution (and not optimal dual solution), we may have to send positive flow along a path (i,k) even if $SP_{ik} > 0$. But the heuristic so described tries to minimize DN_{ik} and hence keep complementary slackness violations as low as possible to get good primal solution. If at the end of execution of algorithm DN = 0, then we have the optimal primal solution. In this way DN_{ik} is similar to Kilter number (ref OUT-OF-KILTER algorithm (a primal dual approach) [13] for solving general min-cost-flow problem).

We find shortest path from every source node 'i' to every sink node 'k' using these slacks as weights, and then make the allocations according to shortest path available. Detailed heuristic is described as under.

Step 0: $X_{ij} = X_{jk} = 0$, $S_{i1k1} = 0$ $\forall$all i, j and k.

Step 1: Compute $S_{ik} = S_{so} + S_{si}$ $\forall$all j and particular i and k.

$$S_{so} = |Csot_{ij} - V_{sso} + U_i + W_j|, \quad S_{si} = |Ctsi_{jk} - V_{ssi} + V_k - W_j|$$

And $S_{ik'} = S_{so'} + S_{si'}$ $\forall j' \neq j$ for the same i, k.

If $S_{ik'} < S_{ik}$ then $S_{i1k1} = S_{ik'}$, Repeat the step $\forall i, j$ and k.

Step 2: Find i and k: $d_k > 0$ and $b_i > 0$.

If $S_{i1k1} < S_{i2k2}$: $i1 \neq i2$ or $k1 \neq k$ $\forall$all i and k then $X_{ij} = X_{jk} = S_{i1k1} = \min(b_i, d_k) = a^*$, $b_i = b_i - a^*$, $d_k = d_k - a^*$.

Step 3: Stop.

Result 3: Heuristic H2 runs in O(n2) time.

Proof: Complexity is dominated by the step 1 which is sorting and can be solved in $O(n^2)$.

5. Results and Discussion

We have solved 150 random problems of varying sizes using methods proposed in this article and the ones developed by Sharma and Saxena [1]. We performed one tail paired-test and F-test on the results. Results of paired t-test are as follows. In terms of duality gap, Subroutine S3($O(n^3)$) performs better than subroutine S2(O(n2)) with the statistical significance of 0.00722 (p-value) in Sharma and saxena [1]. Similarly in terms of duality gap, H2($O(n^3)$) in this paper performs better than H1($O(n^2)$) with a statistical significance of 0.000419 (p-value). H2($O(n^3)$) in this article performs better than S3($O(n^2)$) form Sharma and saxena [1] with a statistical significance of 2.94E−15. For F-test, F-statistic was calculated to be 58.31 as against the f-critical value of 2.62. P-value was calculated to be 4.05E−32. In terms of computational time, no significant difference is registered between these methods, however methods in this paper perform slightly better than those proposed in Sharma and saxena [1]. This is largely due to the fact that we calculate shortest path between the source nodes and sink nodes in contrast to shortest path individually between source and transshipment nodes and transshipment nodes and sink nodes respectively. This method is better computationally.

6. Conclusion

In this work we have developed computationally efficient dual based method to achieve good solution to un-capacitated transshipment problem. As stated earlier, available primal and dual based approaches are capable of solving un-capacitated transshipment problem in $O(n^3 \log(n))$ and $O(m^3 \log(n))$ time respectively. Computational complexities of H1, H2 and H3 are $O(n^2)$, $O(n^3)$ and $O(n^2)$ respectively. Later we intend to extend this work to General Minimum Cost Flow Problem which would have additional capacity constraints on the arcs.

References

[1] Sharma, R.R.K. and Saxena, A. (2002) Dual Based Procedures for the Special Case of Transshipment Problem. *Operation Research*, **39**, 177-188.

[2] Weintraub, A. (1974) A Primal Algorithm to Solve Network Flow Problems with Convex Costs. *Management Science*, **21**, 87-97. https:/doi.org/10.1287/mnsc.21.1.87

[3] Plotkin, S.A. and Tardos, E. (1990) Improved Dual Network Simplex. *Proceedings of the* 1*st Annual ACM-SIAM Symposium on Discrete Algorithms*, Society for Industrial and Applied Mathematics, San Francisco, 22-24 January 1990, 367-376.

[4] Ahuja, R.K. (1993) Network Flows. PhD Thesis, Technische Hochshule Darmstadt, Darmstadt.

[5] Juman, Z.A.M.S. and Hoque, M.A. (2015) An Efficient Heuristic to Obtain a Better Initial Feasible Solution to the Transportation Problem. *Applied Soft Computing*, **34**, 813-826. https://doi.org/10.1016/j.asoc.2015.05.009

[6] Busaker, R.G. and Gowen, P.J. (1961) A Procedure for determining Minimal-Cost Flow Network Patterns. Tech. Rep. ORO-15, Operational Research Office, Johns Hopkins University, Baltimore.

[7] Edmonds, J. and Karp, R.M. (1972) Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems. *Association for Computing Machinery Journal*, **19**, 248-264.

[8] Helgason, R.V. and Kennington, J.L. (1977) An Efficient Procedure for Implementing a Dual Simplex Network Flow Algorithm. *AIIE Transactions*, **9**, 63-68. https://doi.org/10.1080/05695557708975122

[9] Orlin, J.B. (1984) Genuinely Polynomial Simplex and Non-Simplex Algorithms for Minimum Cost Problems. Technical Report 1615-84, Sloan School of Management, MIT, Cambridge, MA.

[10] Ali, A.I., Padman, R. and Thiagarajan, H. (1989) Dual Algorithms for Pure Network Problems. *Operations Research*, **37**, 159-171. https://doi.org/10.1287/opre.37.1.159

[11] Sharma, R.R.K. and Sharma, K.D. (2000) A New Dual Based Procedure for the Transportation Problem. *European Journal of Operational Research*, **122**, 611-624. https://doi.org/10.1016/S0377-2217(99)00081-8

[12] Sharma, R.R.K. and Muralidhar, A. (2009) A New Formulation and Relaxation of the Simple Plant Location Problem. *Asia-Pacific Journal of Operational Research*, **26**, 1-11. https://doi.org/10.1142/S0217595909002122

[13] Clasen, R.J. (1968) The Numerical Solution of Network Problems Using the Out-of-Kilter Algorithm. No. RM-5456-PR. RAND CORP Santa Monica.

Appendix 1

Results of 150 Problems.

Sno	Optimal Solution	Sinha & Sharma Methods						Sharma &Saxena Methods			
		H1		H2		H3		S2		S3	
		Sol	Time	Sol	Time	Sol	Time	Sol	Time	Sol	Time
1	19.3	19.01	0.303	19.2	0.489	19.44	0.096	18.62	0.331	18.82	0.489
2	21.4	21.4	0.313	21.4	0.506	21.56	0.1	21.12	0.342	21.34	0.506
3	17.6	17.56	0.306	17.6	0.494	17.73	0.097	17.21	0.334	17.41	0.494
4	19.8	19.72	0.3	19.76	0.484	19.95	0.095	19.66	0.328	19.68	0.484
5	22.1	21.81	0.287	22.03	0.464	22.26	0.091	20.66	0.314	20.88	0.464
6	15.4	15.4	0.309	15.4	0.5	15.51	0.099	15.22	0.338	15.37	0.5
7	17.3	17.27	0.363	17.27	0.535	17.43	0.105	17.2	0.362	17.23	0.535
8	18.7	18.7	0.291	18.7	0.47	18.84	0.093	18.7	0.318	18.7	0.47
9	19.1	18.79	0.3	18.85	0.484	19.24	0.095	17.5	0.328	17.67	0.484
10	20.4	19.97	0.313	20.24	0.506	20.55	0.1	19.22	0.342	19.42	0.506
11	24.3	23.91	0.329	24.11	0.543	24.48	0.107	23.47	0.347	23.69	0.543
12	17	17	0.331	17	0.551	17.13	0.109	16.78	0.362	16.93	0.551
13	22.5	22.41	0.324	22.41	0.553	22.67	0.109	22.01	0.35	22.21	0.553
14	21.7	21.66	0.318	21.7	0.538	21.86	0.106	21.4	0.344	21.44	0.538
15	25.1	24.77	0.327	24.97	0.561	25.28	0.111	24.3	0.357	24.52	0.561
16	20.5	20.5	0.327	20.5	0.554	20.65	0.109	20.5	0.354	20.5	0.554
17	27.2	27.09	0.351	27.15	0.595	27.4	0.117	26.6	0.38	26.85	0.595
18	28.9	28.9	0.322	28.9	0.544	29.11	0.107	28.9	0.348	28.9	0.544
19	24.2	23.79	0.314	23.98	0.523	24.38	0.103	22.39	0.344	23.57	0.523
20	24.5	23.96	0.332	24.13	0.562	24.68	0.111	22.88	0.359	23.74	0.562
21	24.2	23.76	3.012	24.01	4.403	24.38	0.581	23.35	3.389	23.5	4.403
22	21.5	21.48	3.03	21.5	4.43	21.66	0.567	21.11	3.409	21.24	4.43
23	25.3	25.07	2.966	25.2	4.336	25.49	0.599	24.64	3.337	24.79	4.336
24	21.1	21.02	2.911	21.1	4.256	21.26	0.626	20.66	3.275	20.78	4.256
25	19.9	19.34	2.994	19.12	4.376	20.05	0.653	18.65	3.368	18.75	4.376
26	22.7	22.7	2.994	22.7	4.376	22.87	0.631	22.31	3.368	22.45	4.376
27	23.5	23.45	3.213	23.5	4.698	23.67	0.621	23.05	3.615	23.19	4.698
28	23.9	23.9	2.948	23.9	4.309	24.08	0.644	23.9	3.317	23.9	4.309
29	21.5	21.01	2.875	21.16	4.202	21.66	0.639	20.23	3.234	20.34	4.202
30	24.2	23.06	3.04	23.55	4.443	24.38	0.686	22.7	3.42	22.82	4.443
31	23.8	23.3	3.177	23.54	4.644	23.98	0.628	22.9	3.574	23.04	4.644
32	25.2	25.15	3.315	25.2	4.845	25.39	0.621	24.72	3.729	24.87	4.845

Continued

33	27.4	27.1	3.204	27.1	4.684	27.6	0.648	26.63	3.605	26.8	4.684
34	21.8	21.58	3.149	21.63	4.603	21.96	0.547	21.21	3.543	21.34	4.603
35	24.6	24.08	3.268	24.33	4.777	24.78	0.565	23.67	3.677	23.81	4.777
36	22.9	22.9	3.241	22.9	4.737	23.07	0.552	22.56	3.646	22.69	4.737
37	23.8	23.54	3.479	23.78	5.085	23.98	0.541	23.13	3.914	23.28	5.085
38	26.1	26.1	3.186	26.1	4.657	26.29	0.518	26.1	3.584	26.1	4.657
39	25.3	24.69	3.149	24.95	4.603	25.49	0.558	24.26	3.543	24.41	4.603
40	24.7	23.98	3.287	24.23	4.805	24.88	0.655	23.56	3.698	23.71	4.805
41	25.5	25.17	2.774	25.42	4.055	25.69	0.525	24.74	3.121	24.89	4.055
42	21.2	21.2	2.866	21.2	4.188	21.36	0.541	20.88	3.224	21.01	4.188
43	21.3	21.3	2.802	21.3	4.095	21.46	0.565	20.94	3.152	21.07	4.095
44	23.2	23.15	2.747	23.15	4.015	23.37	0.597	22.76	3.09	22.9	4.015
45	20.7	20.7	2.628	20.7	3.841	20.85	0.955	20.12	2.956	20.24	3.841
46	21.6	21.6	2.829	21.6	4.135	21.76	0.923	21.28	3.183	21.41	4.135
47	22.8	22.62	3.324	22.71	4.858	22.97	0.907	22.41	3.739	22.55	4.858
48	20.8	20.7	2.664	20.76	3.894	20.95	0.941	20.49	2.997	20.61	3.894
49	22.9	22.58	2.747	22.83	4.015	23.07	0.934	22.19	3.09	22.33	4.015
50	22.5	22.07	2.866	22.3	4.188	22.67	1.002	21.69	3.224	21.83	4.188
51	23.8	23.09	3.03	23.32	4.43	23.98	0.918	22.68	3.409	22.82	4.43
52	22.1	21.92	3.132	22.01	4.577	22.26	0.907	21.55	3.523	21.68	4.577
53	22.5	22.12	3.058	22.34	4.47	22.67	0.947	21.74	3.44	21.87	4.47
54	23.1	23.1	3.003	23.1	4.389	23.27	0.799	23.1	3.378	23.1	4.389
55	21.8	21.34	2.875	21.54	4.202	21.96	0.825	20.14	3.234	20.25	4.202
56	26.7	26.51	3.094	26.59	4.523	26.9	0.807	26.06	3.481	26.22	4.523
57	25.3	25.27	3.315	25.27	4.845	25.49	0.791	24.84	3.729	25	4.845
58	27.6	27.6	2.911	27.6	4.256	27.8	0.757	27.6	3.275	27.6	4.256
59	27.1	26.5	3.003	26.8	4.389	27.3	0.815	25.07	3.378	25.2	4.389
60	24.9	24.63	3.132	23.85	4.577	25.08	0.957	23.18	3.523	23.38	4.577
61	25.5	24.76	3.395	25.02	4.961	25.69	0.767	24.33	3.819	24.48	4.961
62	26.4	26.4	3.492	26.4	5.105	26.59	0.791	25.74	3.929	25.9	5.105
63	24.6	24.18	3.458	24.43	5.054	24.78	0.825	23.76	3.89	23.91	5.054
64	21.5	21.31	3.365	21.39	4.919	21.66	1.565	20.94	3.786	21.07	4.919
65	23.9	23.25	3.51	23.49	5.13	24.08	1.574	22.85	3.949	22.99	5.13
66	23.2	23.2	3.464	23.2	5.062	23.37	1.541	22.92	3.897	23.06	5.062
67	24.8	24.38	3.718	24.63	5.435	24.98	1.513	23.96	4.183	24.11	5.435
68	24.7	24.7	3.4	24.7	4.97	24.88	1.555	24.7	3.825	24.7	4.97
69	22.6	22.08	3.302	21.92	4.827	22.77	1.555	21.33	3.715	21.45	4.827
70	22.1	21.48	3.51	21.64	5.13	22.26	1.67	20.71	3.949	20.82	5.13
71	21.6	21.41	6.218	21.56	7.941	21.76	1.532	21.06	6.599	21.19	7.941
72	22.8	22.21	6.256	22.44	7.989	22.97	1.494	21.84	6.639	21.98	7.989

Continued

73	20.8	20.74	6.123	20.8	7.82	20.95	1.579	20.55	6.498	20.68	7.82
74	22.9	22.85	6.011	22.85	7.676	23.07	1.651	22.58	6.379	22.63	7.676
75	22.5	22.34	6.18	22.48	7.892	22.67	1.722	21.98	6.558	22.12	7.892
76	23.8	23.78	6.18	23.8	7.892	23.98	1.665	23.4	6.558	23.54	7.892
77	22.1	21.95	6.634	22.03	8.472	22.26	1.636	21.83	7.04	21.88	8.472
78	22.5	22.5	6.085	22.5	7.772	22.67	1.698	22.28	6.458	22.34	7.772
79	23.1	22.87	5.935	23.01	7.579	23.27	1.684	22.5	6.298	22.64	7.579
80	21.8	21.47	6.274	21.67	8.013	21.96	1.222	21.12	6.658	21.26	8.013
81	26.7	26.59	6.558	26.65	8.375	26.9	1.17	26.33	6.96	26.49	8.375
82	25.3	25.12	6.842	25.27	8.738	25.49	1.259	24.72	7.261	24.87	8.738
83	27.6	27.43	6.615	27.52	8.447	27.8	1.348	26.99	7.02	27.16	8.447
84	27.1	26.64	6.501	26.77	8.302	27.3	1.185	26.21	6.899	26.37	8.302
85	24.9	24.9	6.747	24.9	8.616	25.08	1.222	24.65	7.16	24.85	8.616
86	25.5	25.32	6.689	25.47	8.543	25.69	1.274	24.91	7.099	25.07	8.543
87	26.4	26.4	7.181	26.4	9.171	26.59	1.381	26.14	7.621	26.4	9.171
88	24.6	24.11	6.577	24.38	8.4	24.78	1.421	23.71	6.98	23.86	8.4
89	21.5	20.98	6.501	21.09	8.302	21.66	1.407	20.64	6.899	20.77	8.302
90	23.9	23.76	6.786	23.83	8.666	24.08	1.369	23.37	7.201	23.52	8.666
91	21.6	21.3	5.726	21.47	7.313	21.76	1.428	19.59	6.077	19.7	7.313
92	22.8	22.39	5.915	22.5	7.553	22.97	1.409	21.27	6.277	21.39	7.553
93	20.8	20.24	5.783	20.34	7.386	20.95	1.513	19.91	6.137	20.03	7.386
94	22.9	22.74	5.67	22.9	7.241	23.07	1.383	22.37	6.017	22.51	7.241
95	22.5	22.14	5.424	22.25	6.926	22.67	1.343	21.78	5.756	21.92	6.926
96	23.8	23.61	5.84	23.8	7.459	23.98	1.428	23.23	6.198	23.37	7.459
97	22.1	21.55	6.86	21.66	8.76	22.26	1.513	21.19	7.28	21.33	8.76
98	22.5	21.87	5.499	21.98	7.023	22.67	1.555	21.51	5.836	21.65	7.023
99	23.1	22.94	5.67	23.08	7.241	23.27	1.555	22.57	6.017	22.71	7.241
100	21.8	21.45	5.915	21.56	7.553	21.96	1.67	21.1	6.277	21.23	7.553
101	26.7	26.7	6.256	26.7	7.989	26.9	1.532	26.25	6.639	26.7	7.989
102	25.3	24.74	6.464	24.97	8.255	25.49	1.494	23.43	6.86	23.55	8.255
103	27.6	27.43	6.313	27.52	8.061	27.8	1.579	26.99	6.699	27.16	8.061
104	27.1	27.02	6.199	27.05	7.917	27.3	1.651	26.67	6.579	26.83	7.917
105	24.9	24.9	5.935	24.9	7.579	25.08	1.722	24.5	6.298	24.9	7.579
106	25.5	25.5	6.388	25.5	8.158	25.69	1.665	23.87	6.779	24.02	8.158
107	26.4	26.32	6.842	26.35	8.738	26.59	1.636	26.08	7.261	26.24	8.738
108	24.6	24.5	6.011	24.58	7.676	24.78	1.698	24.26	6.379	24.4	7.676
109	21.5	21.5	6.199	21.5	7.917	21.66	1.684	21.29	6.579	21.5	7.917
110	23.9	23.52	6.464	23.64	8.255	24.08	1.807	21.89	6.86	22.01	8.255
111	21.6	20.69	7.006	21.32	8.948	21.76	1.655	20.37	7.435	20.5	8.948

Continued

112	22.8	22.37	7.211	22.48	9.209	22.97	1.636	22	7.652	22.14	9.209
113	20.8	20.72	7.138	20.8	9.115	20.95	1.708	20.51	7.575	20.63	9.115
114	22.9	22.74	6.947	22.88	8.872	23.07	1.441	22.37	7.372	22.51	8.872
115	22.5	22.37	7.245	22.43	9.253	22.67	0.67	22.01	7.689	22.14	9.253
116	23.8	23.4	7.149	23.51	9.13	23.98	0.651	23.01	7.587	23.16	9.13
117	22.1	22.1	7.675	22.1	9.802	22.26	0.692	21.88	8.145	22.01	9.802
118	22.5	22.43	7.019	22.46	8.964	22.67	1.225	21.98	7.449	22.12	8.964
119	23.1	22.98	6.817	23.03	8.706	23.27	1.233	22.87	7.234	22.89	8.706
120	21.8	21.45	7.245	21.56	9.253	21.96	1.207	21.02	7.689	21.15	9.253
121	26.7	26.49	6.501	26.46	6.746	26.9	1.185	26.38	6.61	26.54	6.61
122	25.3	25.17	6.747	24.79	7.002	25.49	1.218	24.95	6.86	24.59	6.86
123	27.6	27.21	6.689	27.49	6.941	27.8	1.218	26.97	6.801	27.6	6.801
124	27.1	26.59	7.181	26.91	7.452	27.3	1.307	26.64	7.302	25.72	7.302
125	24.9	24.15	6.577	24.9	6.825	25.08	1.199	24.6	6.688	23.46	6.688
126	25.5	25.3	6.501	25.19	6.746	25.69	1.17	25.25	6.61	25.02	6.61
127	26.4	25.95	6.786	26.29	7.042	26.59	1.236	25.71	6.9	25.45	6.9
128	24.6	24.6	5.726	24.58	5.942	24.78	1.292	23.84	5.822	24.45	5.822
129	21.5	21.05	5.915	21.5	6.138	21.66	1.348	21.2	6.014	21.24	6.014
130	23.9	23.73	5.783	23.64	6.001	24.08	1.304	23.35	5.88	23.49	5.88
131	21.6	21.58	5.67	20.69	5.884	21.76	1.281	21.12	5.765	21.36	5.765
132	22.8	22.8	5.424	22.37	5.629	22.97	1.33	22.05	5.515	22.57	5.515
133	20.8	20.34	5.84	20.8	6.06	20.95	0.628	20.59	5.938	20.65	5.938
134	22.9	22.65	6.86	22.74	7.119	23.07	0.621	22.37	6.975	22.44	6.975
135	26.7	25.93	5.499	26.57	5.707	26.9	0.648	26.43	5.591	26.03	5.591
136	25.3	25.3	5.67	24.87	5.884	25.49	0.547	24.39	5.765	25.1	5.765
137	27.6	27.13	5.915	27.6	6.138	27.8	0.565	26.5	6.014	27.13	6.014
138	27.1	26.86	6.256	26.91	6.492	27.3	0.552	26.5	6.361	26.67	6.361
139	24.9	24.23	3.003	24.9	3.116	25.08	0.541	22.58	3.053	24.23	3.053
140	25.5	25.5	2.875	24.74	2.983	25.69	0.518	23.79	2.923	25.45	2.923
141	26.4	25.95	3.094	25.85	3.211	26.59	0.558	25.26	3.146	25.95	3.146
142	24.6	24.6	3.315	24.55	3.44	24.78	0.655	24.03	3.371	24.6	3.371
143	21.5	21.01	2.911	21.16	3.021	21.66	0.525	20.81	2.96	20.86	2.96
144	23.9	23.23	3.003	23.9	3.116	24.08	0.541	23.33	3.053	23.09	3.053
145	23.2	22.99	3.132	23.11	3.25	23.37	0.565	22.25	3.185	22.81	3.185
146	24.8	24.16	3.395	24.8	3.523	24.98	0.597	23.71	3.452	24.16	3.452
147	24.7	24.63	3.492	24.58	3.624	24.88	0.617	24.13	3.551	24.26	3.551
148	22.6	22.55	3.458	22.6	3.588	22.77	0.603	21.88	3.516	21.81	3.516
149	22.1	21.95	3.365	22.06	3.492	22.26	0.592	21.72	3.422	21.26	3.422
150	21.6	21.58	3.51	21.6	3.642	21.76	0.998	20	3.569	21.23	3.569