# Appendix A
# Thermo-physical Properties of the S355 Grade Steel

| T (°C) | Fraction liquid (wt%) | Density (g/cm$^3$) | Thermal conductivity [W/(m K)] | Electrical resistivity (10e-6 Ohm m) | Young's modulus (GPa) |
|---|---|---|---|---|---|
| 1600 | 100 | 6.91218 | 34.91665 | 1.31434 | 0 |
| 1580 | 100 | 6.92858 | 34.54952 | 1.31412 | 0 |
| 1560 | 100 | 6.94487 | 34.18239 | 1.3139 | 0 |
| 1540 | 100 | 6.96107 | 33.81526 | 1.31367 | 0 |
| 1520 | 100 | 6.97717 | 33.44813 | 1.31344 | 0 |
| 1513.7 | 99.99996 | 6.98222 | 33.33249 | 1.31337 | 1.13E-24 |
| 1500 | 46.85386 | 7.13721 | 36.00007 | 1.20672 | 6.38733 |
| 1486.24 | 25.80748 | 7.20396 | 37.09981 | 1.16187 | 25.52687 |
| 1483.93 | 10.7205 | 7.25349 | 35.0414 | 1.2285 | 46.73048 |
| 1480 | 6.46744 | 7.2678 | 35.1107 | 1.22334 | 57.04138 |
| 1465.4 | 0.19412 | 7.29098 | 35.10308 | 1.21341 | 76.17767 |
| 1465.22 | 4.91E-05 | 7.29142 | 35.10561 | 1.2132 | 76.81795 |
| 1460 | | 7.29426 | 35.03964 | 1.21183 | 77.25785 |
| 1420 | | 7.31594 | 34.53488 | 1.20117 | 80.62455 |
| 1400 | | 7.32673 | 34.28274 | 1.19571 | 82.30757 |
| 1360 | | 7.34827 | 33.77882 | 1.18453 | 85.67304 |
| 1340 | | 7.35903 | 33.52701 | 1.17882 | 87.35554 |
| 1300 | | 7.38052 | 33.02361 | 1.16711 | 90.72018 |
| 1280 | | 7.39127 | 32.772 | 1.16112 | 92.40234 |
| 1240 | | 7.41278 | 32.26891 | 1.14885 | 95.76642 |
| 1200 | | 7.43433 | 31.76595 | 1.13619 | 99.13026 |
| 1180 | | 7.44512 | 31.5145 | 1.12971 | 100.81211 |
| 1140 | | 7.46675 | 31.01165 | 1.11642 | 104.17572 |
| 1120 | | 7.47759 | 30.76024 | 1.10962 | 105.85749 |
| 1080 | | 7.49932 | 30.25745 | 1.09567 | 109.22097 |

(continued)



M. Hojny, *Modeling Steel Deformation in the Semi-Solid State*,
Advanced Structured Materials 47, DOI 10.1007/978-3-319-67976-1

(continued)

| T (°C) | Fraction liquid (wt%) | Density (g/cm$^3$) | Thermal conductivity [W/(m K)] | Electrical resistivity (10e-6 Ohm m) | Young's modulus (GPa) |
|---|---|---|---|---|---|
| 1040 | | 7.52113 | 29.75469 | 1.08125 | 112.5844 |
| 1020 | | 7.53206 | 29.50331 | 1.07385 | 114.2661 |
| 980 | | 7.55399 | 29.00056 | 1.05868 | 117.6295 |
| 940 | | 7.57599 | 28.49781 | 1.04296 | 120.99288 |
| 900 | | 7.59809 | 27.99507 | 1.02669 | 124.35625 |
| 880 | | 7.60917 | 27.7437 | 1.01833 | 126.03793 |
| 840 | | 7.6314 | 27.24096 | 1.00115 | 129.4013 |
| 822.87 | | 7.64095 | 27.02563 | 0.99359 | 130.84191 |
| 820 | | 7.63921 | 27.19745 | 0.98473 | 131.75148 |
| 780 | | 7.62239 | 29.24646 | 0.88223 | 141.69614 |
| 760 | | 7.61956 | 30.00841 | 0.8435 | 145.34667 |
| 720 | | 7.62059 | 31.21428 | 0.77952 | 151.26621 |
| 706.93 | | 7.62221 | 31.54512 | 0.7612 | 152.93539 |
| 700 | | 7.61719 | 32.55451 | 0.73238 | 155.69517 |
| 693.09 | | 7.61436 | 33.32865 | 0.71029 | 157.88905 |
| 660 | | 7.62712 | 33.61721 | 0.68007 | 160.57609 |
| 640 | | 7.63474 | 33.8191 | 0.66152 | 162.19246 |
| 600 | | 7.64982 | 34.2856 | 0.62394 | 165.40907 |
| 580 | | 7.65727 | 34.55049 | 0.60497 | 167.00971 |
| 540 | | 7.67204 | 35.14422 | 0.56687 | 170.19637 |
| 520 | | 7.67936 | 35.47328 | 0.5478 | 171.78252 |
| 480.87 | | 7.69356 | 36.1795 | 0.5106 | 174.87229 |
| 480 | | 7.69389 | 36.20218 | 0.5097 | 174.94655 |
| 440 | | 7.70932 | 37.31181 | 0.46827 | 178.40557 |
| 431.86 | | 7.71246 | 37.54104 | 0.4601 | 179.09951 |
| 420 | | 7.71658 | 37.80282 | 0.44923 | 180.02249 |
| 380 | | 7.73035 | 38.74111 | 0.41305 | 183.11825 |
| 360 | | 7.73716 | 39.24242 | 0.39529 | 184.65566 |
| 320 | | 7.7506 | 40.3099 | 0.36051 | 187.70846 |
| 300 | | 7.75724 | 40.87627 | 0.34353 | 189.2233 |
| 260 | | 7.77033 | 42.07472 | 0.31045 | 192.22824 |
| 240 | | 7.77678 | 42.70698 | 0.29438 | 193.71764 |
| 200 | | 7.7895 | 44.03823 | 0.26323 | 196.66875 |
| 180 | | 7.79577 | 44.73759 | 0.24816 | 198.13017 |
| 140 | | 7.80812 | 46.2054 | 0.21907 | 201.02563 |
| 120 | | 7.81421 | 46.97471 | 0.20505 | 202.46062 |
| 80 | | 7.82621 | 48.58644 | 0.17808 | 205.30793 |
| 68.52 | | 7.82962 | 49.06718 | 0.1706 | 206.1196 |

(continued)

(continued)

| T (°C) | Fraction liquid (wt%) | Density (g/cm$^3$) | Thermal conductivity [W/(m K)] | Electrical resistivity (10e-6 Ohm m) | Young's modulus (GPa) |
|---|---|---|---|---|---|
| 40 | | 7.83811 | 50.27003 | 0.15262 | 208.11896 |
| 25 | | 7.84248 | 50.93719 | 0.14341 | 209.1696 |

| T (°C) | Poisson's ratio | Liquid viscosity (mPa s) | Enthalpy (J/g) | Specific heat [J/(g K)] | Latent heat (J/g) |
|---|---|---|---|---|---|
| 1600 | 0.49992 | 5.66944 | 1341.7645 | 0.82688 | |
| 1580 | 0.49992 | 5.84298 | 1325.2278 | 0.82684 | |
| 1560 | 0.49992 | 6.0258 | 1308.6915 | 0.82682 | |
| 1540 | 0.49992 | 6.21856 | 1292.1556 | 0.82679 | |
| 1520 | 0.49992 | 6.42199 | 1275.7131 | 0.81806 | |
| 1513.7 | 0.49992 | 6.48841 | 1270.5687 | | 50.97787 |
| 1500 | 0.42097 | 6.44337 | 1131.8002 | 6.05028 | 96.93005 |
| 1486.24 | 0.3882 | 6.40566 | 1072.0753 | | 128.23126 |
| 1483.93 | 0.39375 | 6.40141 | 1015.0818 | | 150.54388 |
| 1480 | 0.38815 | 6.38227 | 1001.5291 | 2.97105 | 167.01298 |
| 1465.4 | 0.37924 | 6.32939 | 975.70594 | | 211.36935 |
| 1465.22 | 0.37898 | 6.32857 | 975.06483 | | 241.82898 |
| 1460 | 0.37868 | | 971.38423 | 0.70351 | |
| 1420 | 0.37633 | | 943.38307 | 0.69556 | |
| 1400 | 0.37515 | | 929.49892 | 0.69173 | |
| 1360 | 0.3728 | | 901.9554 | 0.68431 | |
| 1340 | 0.37162 | | 888.29294 | 0.6807 | |
| 1300 | 0.36925 | | 861.18047 | 0.67365 | |
| 1280 | 0.36807 | | 847.72861 | 0.67016 | |
| 1240 | 0.36571 | | 821.03032 | 0.66333 | |
| 1200 | 0.36334 | | 794.59762 | 0.65679 | |
| 1180 | 0.36216 | | 781.47837 | 0.65355 | |
| 1140 | 0.35979 | | 755.43174 | 0.64711 | |
| 1120 | 0.35861 | | 742.50364 | 0.64393 | |
| 1080 | 0.35624 | | 716.83646 | 0.63761 | |
| 1040 | 0.35387 | | 691.41987 | 0.63129 | |
| 1020 | 0.35269 | | 678.80514 | 0.62816 | |
| 980 | 0.35032 | | 653.76217 | 0.62191 | |
| 940 | 0.34796 | | 628.96727 | 0.61568 | |
| 900 | 0.34559 | | 604.41988 | 0.60943 | |

(continued)

(continued)

| T (°C) | Poisson's ratio | Liquid viscosity (mPa s) | Enthalpy (J/g) | Specific heat [J/(g K)] | Latent heat (J/g) |
|---|---|---|---|---|---|
| 880 | 0.3444 | | 592.23888 | 0.60632 | |
| 840 | 0.34204 | | 568.06213 | 0.60011 | |
| 822.87 | 0.34102 | | 557.78226 | | |
| 820 | 0.3401 | | 554.9352 | 0.99814 | |
| 780 | 0.32991 | | 514.98116 | 0.99243 | |
| 760 | 0.32655 | | 495.06587 | 1.00289 | |
| 720 | 0.32173 | | 452.16563 | 1.0585 | |
| 706.93 | 0.32051 | | 438.48967 | | |
| 700 | 0.31745 | | 421.67167 | 2.16441 | |
| 693.09 | 0.3153 | | 408.16114 | | |
| 660 | 0.31401 | | 377.71663 | 0.8885 | |
| 640 | 0.31324 | | 360.24442 | 0.85654 | |
| 600 | 0.3117 | | 327.04394 | 0.80251 | |
| 580 | 0.31093 | | 311.19806 | 0.77943 | |
| 540 | 0.30939 | | 280.78854 | 0.73914 | |
| 520 | 0.30862 | | 266.15472 | 0.72132 | |
| 480.87 | 0.30712 | | 238.49024 | | |
| 480 | 0.30708 | | 237.85883 | 0.72682 | |
| 440 | 0.3052 | | 209.02962 | 0.70229 | |
| 431.86 | 0.30482 | | 203.33605 | | |
| 420 | 0.30436 | | 195.61239 | 0.64602 | |
| 380 | 0.30281 | | 170.2417 | 0.62038 | |
| 360 | 0.30203 | | 157.93091 | 0.6083 | |
| 320 | 0.30047 | | 134.01322 | 0.5859 | |
| 300 | 0.29969 | | 122.39202 | 0.57492 | |
| 260 | 0.29813 | | 99.79758 | 0.55366 | |
| 240 | 0.29735 | | 88.81615 | 0.54331 | |
| 200 | 0.29579 | | 67.46756 | 0.52339 | |
| 180 | 0.29501 | | 57.09232 | 0.51364 | |
| 140 | 0.29345 | | 36.9104 | 0.49522 | |
| 120 | 0.29267 | | 27.08885 | 0.48655 | |
| 80 | 0.29112 | | 7.95299 | 0.46994 | |
| 68.52 | 0.29067 | | 2.58538 | | |
| 40 | 0.28956 | | | 0.45329 | |
| 25 | 0.28897 | | | 0.44669 | |

# Appendix B
# Thermo-physical Properties of the C45 Grade Steel

| T (°C) | Fraction liquid (wt) | Density (g/cm$^3$) | Thermal conductivity [W/(m K)] | Electrical resistivity (10e-6 Ohm m) | Young's modulus (GPa) |
|---|---|---|---|---|---|
| 1600 | 1 | 6.91435 | 35.34082 | 1.29856 | 0 |
| 1550 | 1 | 6.95495 | 34.4199 | 1.29771 | 0 |
| 1505 | 1 | 6.99096 | 33.59108 | 1.29691 | 0 |
| 1500 | 1 | 6.99493 | 33.49898 | 1.29682 | 0 |
| 1495 | 1 | 6.99889 | 33.40689 | 1.29673 | 0 |
| 1494.79 | 1 | 6.99906 | 33.40304 | 1.29672 | 0 |
| 1490 | 0.87841 | 7.03448 | 33.97108 | 1.27159 | 0.0165101 |
| 1489.88 | 0.87555 | 7.03532 | 33.9845 | 1.271 | 0.0181056 |
| 1489.88 | 0.87553 | 7.03534 | 33.98456 | 1.271 | 0.0181014 |
| 1485 | 0.7273 | 7.07865 | 34.14199 | 1.26163 | 0.0941919 |
| 1480 | 0.60769 | 7.11483 | 34.26067 | 1.25369 | 0.84088 |
| 1475 | 0.51111 | 7.14508 | 34.34632 | 1.24699 | 2.86007 |
| 1470 | 0.43136 | 7.17097 | 34.40652 | 1.24125 | 6.27462 |
| 1465 | 0.36426 | 7.19354 | 34.44713 | 1.23623 | 10.88971 |
| 1460 | 0.30683 | 7.2136 | 34.47117 | 1.23182 | 16.42427 |
| 1455 | 0.25701 | 7.23165 | 34.48333 | 1.22783 | 22.61462 |
| 1450 | 0.21335 | 7.24802 | 34.48592 | 1.22419 | 29.25093 |
| 1445 | 0.17473 | 7.26307 | 34.48068 | 1.22082 | 36.18725 |
| 1440 | 0.14025 | 7.277 | 34.46906 | 1.21768 | 43.32377 |
| 1435 | 0.10925 | 7.28997 | 34.45225 | 1.21472 | 50.59053 |
| 1430 | 0.0811754 | 7.30214 | 34.43121 | 1.2119 | 57.94115 |
| 1425 | 0.0556055 | 7.31363 | 34.40654 | 1.20921 | 65.33844 |
| 1420 | 0.0321876 | 7.32452 | 34.37859 | 1.20663 | 72.758 |
| 1415 | 0.0106378 | 7.33489 | 34.34723 | 1.20416 | 80.18024 |
| 1412.42 | 0.0061221 | 7.34013 | 34.32971 | 1.20293 | 84.06567 |

(continued)



M. Hojny, *Modeling Steel Deformation in the Semi-Solid State*,
Advanced Structured Materials 47, DOI 10.1007/978-3-319-67976-1

(continued)

| T (°C) | Fraction liquid (wt) | Density (g/cm$^3$) | Thermal conductivity [W/(m K)] | Electrical resistivity (10e-6 Ohm m) | Young's modulus (GPa) |
|---|---|---|---|---|---|
| 1405 | | 7.34391 | 34.23823 | 1.20084 | 84.69597 |
| 1400 | | 7.34646 | 34.17663 | 1.19942 | 85.12081 |
| 1370 | | 7.36177 | 33.80816 | 1.19075 | 87.66882 |
| 1340 | | 7.3771 | 33.44172 | 1.18182 | 90.21507 |
| 1310 | | 7.39246 | 33.07744 | 1.17262 | 92.75958 |
| 1280 | | 7.40785 | 32.71543 | 1.16313 | 95.30232 |
| 1250 | | 7.42327 | 32.35579 | 1.15334 | 97.8433 |
| 1220 | | 7.43871 | 31.99862 | 1.14324 | 100.38251 |
| 1190 | | 7.45417 | 31.64402 | 1.13283 | 102.91995 |
| 1160 | | 7.46967 | 31.29205 | 1.12208 | 105.45561 |
| 1130 | | 7.48519 | 30.9428 | 1.11099 | 107.98949 |
| 1100 | | 7.50075 | 30.59631 | 1.09955 | 110.52159 |
| 1070 | | 7.51633 | 30.25262 | 1.08775 | 113.05189 |
| 1040 | | 7.53194 | 29.91177 | 1.07557 | 115.5804 |
| 1000 | | 7.5528 | 29.46171 | 1.05874 | 118.94894 |
| 960 | | 7.57372 | 29.01666 | 1.0412 | 122.31426 |
| 920 | | 7.59469 | 28.57653 | 1.02294 | 125.67635 |
| 900 | | 7.6052 | 28.35825 | 1.01354 | 127.35618 |
| 860 | | 7.62625 | 27.92515 | 0.99416 | 130.71339 |
| 830 | | 7.64208 | 27.60321 | 0.97913 | 133.22916 |
| 800 | | 7.65795 | 27.28359 | 0.96366 | 135.74307 |
| 770 | | 7.67385 | 26.96612 | 0.94775 | 138.25514 |
| 765 | | 7.6765 | 26.91341 | 0.94506 | 138.67363 |
| 760 | | 7.67915 | 26.86075 | 0.94235 | 139.09208 |
| 755 | | 7.68181 | 26.80814 | 0.93963 | 139.51047 |
| 750 | | 7.68446 | 26.75559 | 0.93689 | 139.92881 |
| 745 | | 7.68712 | 26.70308 | 0.93415 | 140.3471 |
| 740 | | 7.68978 | 26.65063 | 0.93139 | 140.76534 |
| 735 | | 7.69244 | 26.59822 | 0.92862 | 141.18352 |
| 730 | | 7.6951 | 26.54587 | 0.92584 | 141.60166 |
| 725 | | 7.69776 | 26.49356 | 0.92304 | 142.01974 |
| 720 | | 7.70042 | 26.4413 | 0.92023 | 142.43777 |
| 715 | | 7.70308 | 26.38908 | 0.91741 | 142.85575 |
| 710 | | 7.70575 | 26.33691 | 0.91458 | 143.27367 |
| 705 | | 7.70841 | 26.28478 | 0.91173 | 143.69154 |
| 700 | | 7.71108 | 26.23269 | 0.90887 | 144.10937 |
| 695 | | 7.71374 | 26.18064 | 0.906 | 144.52713 |
| 690 | | 7.71641 | 26.12863 | 0.90312 | 144.94485 |

(continued)

(continued)

| T (°C) | Fraction liquid (wt) | Density (g/cm$^3$) | Thermal conductivity [W/(m K)] | Electrical resistivity (10e-6 Ohm m) | Young's modulus (GPa) |
|---|---|---|---|---|---|
| 685 | | 7.71908 | 26.07666 | 0.90022 | 145.36252 |
| 680 | | 7.72175 | 26.02473 | 0.89731 | 145.78013 |
| 675 | | 7.72442 | 25.97284 | 0.89438 | 146.19769 |
| 670 | | 7.72709 | 25.92098 | 0.89145 | 146.61519 |
| 665 | | 7.72977 | 25.86915 | 0.8885 | 147.03265 |
| 660 | | 7.73244 | 25.81736 | 0.88553 | 147.45005 |
| 655 | | 7.73511 | 25.7656 | 0.88256 | 147.8674 |
| 650 | | 7.73779 | 25.71387 | 0.87957 | 148.2847 |
| 645 | | 7.74047 | 25.66217 | 0.87657 | 148.70194 |
| 640 | | 7.74314 | 25.6105 | 0.87355 | 149.11913 |
| 635 | | 7.74582 | 25.55885 | 0.87053 | 149.53627 |
| 630 | | 7.7485 | 25.50723 | 0.86749 | 149.95336 |
| 615 | | 7.75655 | 25.35252 | 0.85828 | 151.2043 |
| 610 | | 7.75923 | 25.30099 | 0.85519 | 151.62118 |
| 600 | | 7.7646 | 25.19799 | 0.84896 | 152.45477 |

| T (°C) | Poisson's ratio | Liquid viscosity (mPa s) | Enthalpy (J/g) | Specific heat [J/(g K)] | Latent heat (J/g) |
|---|---|---|---|---|---|
| 1600 | 0.49968 | 5.44574 | 1351.003 | | |
| 1550 | 0.49968 | 5.87455 | 1309.562 | 0.8288 | |
| 1505 | 0.49968 | 6.31224 | 1272.547 | 0.814 | |
| 1500 | 0.49968 | 6.36427 | 1268.489 | 0.8116 | |
| 1495 | 0.49968 | 6.41703 | 1264.442 | 0.8094 | |
| 1494.79 | 0.49968 | 6.41926 | 1264.274 | 0.80371 | |
| 1490 | 0.48196 | 6.40787 | 1229.413 | 7.2764 | 28.871 |
| 1489.88 | 0.48155 | 6.40771 | 1228.587 | | 29.554 |
| 1489.88 | 0.48154 | 6.40771 | 1228.638 | | 29.502 |
| 1485 | 0.46441 | 6.37848 | 1184.669 | 8.9488 | 67.637 |
| 1480 | 0.45033 | 6.35021 | 1148.982 | 7.1374 | 98.081 |
| 1475 | 0.43876 | 6.32358 | 1119.988 | 5.7988 | 122.357 |
| 1470 | 0.42905 | 6.29814 | 1095.888 | 4.82 | 142.121 |
| 1465 | 0.42075 | 6.27418 | 1075.471 | 4.0834 | 158.489 |
| 1460 | 0.41354 | 6.25124 | 1057.88 | 3.5182 | 172.255 |
| 1455 | 0.40719 | 6.22958 | 1042.516 | 3.0728 | 183.969 |
| 1450 | 0.40155 | 6.20922 | 1028.956 | 2.712 | 194.021 |

(continued)

(continued)

| T (°C) | Poisson's ratio | Liquid viscosity (mPa s) | Enthalpy (J/g) | Specific heat [J/(g K)] | Latent heat (J/g) |
|---|---|---|---|---|---|
| 1445 | 0.39648 | 6.18996 | 1016.866 | 2.418 | 202.715 |
| 1440 | 0.39189 | 6.17154 | 1005.992 | 2.1748 | 210.288 |
| 1435 | 0.38771 | 6.15421 | 996.137 | 1.971 | 216.921 |
| 1430 | 0.38386 | 6.13801 | 987.143 | 1.7988 | 222.76 |
| 1425 | 0.38031 | 6.1227 | 978.883 | 1.652 | 227.919 |
| 1420 | 0.37701 | 6.10826 | 971.257 | 1.5252 | 232.494 |
| 1415 | 0.37393 | 6.09474 | 964.18 | 1.4154 | 236.562 |
| 1412.42 | 0.3724 | 6.06324 | 960.689 | 1.35151 | 238.52 |
| 1405 | 0.37197 | | 955.515 | 0.69759 | |
| 1400 | 0.37169 | | 952.036 | 0.6958 | |
| 1370 | 0.36997 | | 931.247 | 0.691 | |
| 1340 | 0.36825 | | 910.595 | 0.6864 | |
| 1310 | 0.36654 | | 890.077 | 0.682 | |
| 1280 | 0.36482 | | 869.693 | 0.6776 | |
| 1250 | 0.3631 | | 849.442 | 0.6732 | |
| 1220 | 0.36139 | | 829.321 | 0.6688 | |
| 1190 | 0.35967 | | 809.328 | 0.6648 | |
| 1160 | 0.35795 | | 789.463 | 0.6604 | |
| 1130 | 0.35624 | | 769.721 | 0.6564 | |
| 1100 | 0.35452 | | 750.102 | 0.6522 | |
| 1070 | 0.3528 | | 730.6 | 0.6484 | |
| 1040 | 0.35109 | | 711.212 | 0.6448 | |
| 1000 | 0.3488 | | 685.531 | 0.64 | |
| 960 | 0.34651 | | 660.03 | 0.6356 | |
| 920 | 0.34422 | | 634.687 | 0.632 | |
| 900 | 0.34308 | | 622.067 | 0.6304 | |
| 860 | 0.34079 | | 596.902 | 0.6284 | |
| 830 | 0.33908 | | 578.066 | 0.6278 | |
| 800 | 0.33736 | | 559.217 | 0.6288 | |
| 770 | 0.33564 | | 540.288 | 0.633 | |
| 765 | 0.33536 | | 537.119 | 0.6338 | |
| 760 | 0.33507 | | 533.878 | 0.6482 | |
| 755 | 0.33479 | | 530.629 | 0.6498 | |
| 750 | 0.3345 | | 527.391 | 0.6476 | |
| 745 | 0.33421 | | 524.166 | 0.645 | |
| 740 | 0.33393 | | 520.952 | 0.6428 | |
| 735 | 0.33364 | | 517.748 | 0.6408 | |
| 730 | 0.33336 | | 514.556 | 0.6384 | |

(continued)

(continued)

| T (°C) | Poisson's ratio | Liquid viscosity (mPa s) | Enthalpy (J/g) | Specific heat [J/(g K)] | Latent heat (J/g) |
|---|---|---|---|---|---|
| 725 | 0.33307 | | 511.374 | 0.6364 | |
| 720 | 0.33278 | | 508.203 | 0.6342 | |
| 715 | 0.3325 | | 505.042 | 0.6322 | |
| 710 | 0.33221 | | 501.891 | 0.6302 | |
| 705 | 0.33193 | | 498.749 | 0.6284 | |
| 700 | 0.33164 | | 495.617 | 0.6264 | |
| 695 | 0.33135 | | 492.495 | 0.6244 | |
| 690 | 0.33107 | | 489.382 | 0.6226 | |
| 685 | 0.33078 | | 486.277 | 0.621 | |
| 680 | 0.3305 | | 483.182 | 0.619 | |
| 675 | 0.33021 | | 480.095 | 0.6174 | |
| 670 | 0.32993 | | 477.017 | 0.6156 | |
| 665 | 0.32964 | | 473.947 | 0.614 | |
| 660 | 0.32935 | | 470.886 | 0.6122 | |
| 655 | 0.32907 | | 467.832 | 0.6108 | |
| 650 | 0.32878 | | 464.787 | 0.609 | |
| 645 | 0.3285 | | 461.75 | 0.6074 | |
| 640 | 0.32821 | | 458.72 | 0.606 | |
| 635 | 0.32792 | | 455.698 | 0.6044 | |
| 630 | 0.32764 | | 452.684 | 0.6028 | |
| 615 | 0.32678 | | 443.686 | 0.5984 | |
| 610 | 0.32649 | | 440.701 | 0.597 | |
| 600 | 0.32592 | | 434.753 | 0.5942 | |

# Appendix C
# Complete Source Code: Steady Heat Flow

As an example, let's consider the area presented in Fig. C.1, which is a cross-section of e.g. a rolled strand with dimensions of $0.5 \times 0.5$ m. The left, right and top surface of the cast strand are heated by the medium with a temperature 500 °C. On the other hand the bottom surface of the cast strand is cooled by the medium with a temperature 20 °C. The heat transfer coefficient for all boundary conditions assumes the same value of $\alpha = 100\left(\mathrm{W}/\mathrm{m}^2\mathrm{K}\right)$, and heat transfer coefficients of $\lambda_x = \lambda_y = 78(\mathrm{W}/\mathrm{mK})$. The power of heat sources referred to the volume unit in the analysed case $q_{vol} = 0(\mathrm{W}/\mathrm{m}^3)$.

Below find a complete print-out of the source code written in the C++ language, simulating the steady state condition of heat transfer in the cross-section. The comments placed within the code will enable the reader to analyse the numerical code individually, and will be helpful during potential modifications implemented by the user, or when creating original solutions. The program code may be

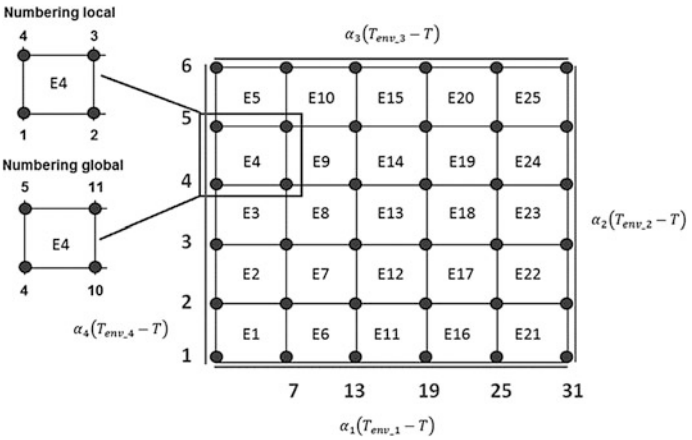


**Fig. C. 1** A cross-section of a cast strand with defined third type boundary conditions



M. Hojny, *Modeling Steel Deformation in the Semi-Solid State*,
Advanced Structured Materials 47, DOI 10.1007/978-3-319-67976-1

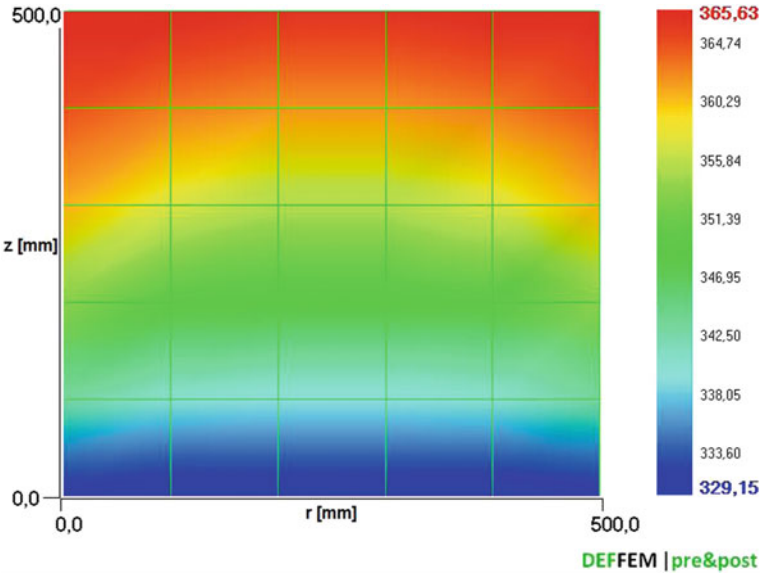


**Fig. C.2** The temperature field distribution for the case analysed (steady state condition)

compiled with any C++ compiler (e.g. Dev-C++ 5.11). The final result will be the temperature field distribution for the steady state condition (test data) (Fig. C.2).

```
#include <iostream>
#include <fstream>


main()
{
float a[220][220],x_w[251],zamiana,stala,b,max;
int nlh,nlv,nw,ne,ie[1000][5];
int l,licznik,k,i,j,u,ii,jj;
float r,h,lambdaX,lambdaY,moc_zrodel,strumien;
float t_czynnika1, t_czynnika2, t_czynnika3, t_czynnika4;
float alfa1,alfa2,alfa3,alfa4;
float dx,dy,dv,dh,wx[1000],wy[1000],x[1000],y[1000],xe[500],ye[500];
float Kc_e[5][5],Kalfa_e[5][5];
float Kc[251][251],Kalfa[251][251],K[251][251],f[251][2];
float fq_duze[251][2],fq_duze_e[5][2],falfa_e[5][2],falfa[251][2];
float x1,x2,x3,x4,y1,y2,y3,y4;
float czlon_lambda_x,czlon_lambda_y,czlon_alfa_bok23,czlon_alfa_bok41;
float czlon_alfa_bok12,czlon_alfa_bok34;
float czlon_fq_duze,czlon_falfa,czlon_q;
int n,imax,spr=1;
```

```
//-------------------------- INPUT PARAMETERS-------------------------
//nlh-the number of horizontal lines in the mesh
//nlv-the number of vertical lines in the mesh
//nw,ne-number of mesh nodes, number of elements
//h,r-height, width section
//dv,dh-increases the length of elements for h and r
//dx,dy-operating variables
//wx[],wy[]-tables containing the coordinates of the mesh
//x[],y[]-coordinates x and y nodes of the mesh
//------------------------------ INPUT DATA-----------------------------
nlh=6;
nlv=6;
h=500;
r=500;
h=h/1000;
r=r/1000;            //conversion of centimeters to meters
lambdaX=78;         //thermal conductivity, x-direction, W/(m*K)
lambdaY=78;         //thermal conductivity, y-direction, W/(m*K)
alfa1=100;           //heat transfer coefficient, bottom section, W/(m2*K)
alfa2=100;           //heat transfer coefficient, right section, W/(m2*K)
alfa3=100;           //heat transfer coefficient, top section, W/(m2*K)
alfa4=100;           //heat transfer coefficient, left section, W/(m2*K)
moc_zrodel=0;       //power sources of heat per unit volume, (W/m3)
t_czynnika1=20;     //
the temperature of the cooling medium, bottom section, [st. C]
t_czynnika2=500;    //
the temperature of the cooling medium, right section, [st. C]
t_czynnika3=500;    //
the temperature of the cooling medium, top section, [st. C]
t_czynnika4=500;    //
the temperature of the cooling medium, left section, [st. C]
//---------------------------------------------------------------------
dx=0;
dy=0;
dv=h/(nlh-1);
dh=r/(nlv-1);
nw=nlh*nlv;
ne=(nlh-1)*(nlv-1);
std::cout<<"Parameters the generated mesh:"<<std::endl;
std::cout<<"Number of elements:"<<" "<<ne<<std::endl;
std::cout<<"Number of nodes:"<<" "<<nw<<std::endl;
std::cout<<std::endl;
std::cout<<"ENTER-START CALCULATION";
std::cin.get();
std::cin.get();
```

```
//---------------- DETERMINATION OF THE MESH COORDINATES ----------------
for(int i=1;i<=nlv;i++)
{
wx[i]=dx;
dx=dx+dh;
}
for(int j=1;j<=nlh;j++)
{
wy[j]=dy;
dy=dy+dv;
}
//--------------- DETERMINING COORDINATES OF A GRID NODES ---------------
licznik=1;
l=0;
for(int i=1;i<=nw;i++)
{
x[i]=wx[licznik];
l++;
if(l==nlh)
{
licznik++;
l=0;
}}
licznik=1;
for(int i=1;i<=nw;i++)
{
y[i]=wy[licznik];
licznik++;
if(licznik==nlh+1)
{
licznik=1;
}}
//----------------- DETERMINATION OF MATRIX CONNECTION -----------------
k=1;
i=1;
for(int w=1;w<=ne;w++)
{
ie[w][1]=i;
ie[w][2]=i+nlh;
ie[w][3]=1+i+nlh;
ie[w][4]=1+i;
k++;
i++;
if(k==nlh)
{
```

```
k=1;
i=i+1;
}}
//-------------------------------------------------------------------------
//Calculation of the thermal conductivity of the matrix
//Kc_e for one single element and global matrix conductivity Kc
//-------------------------------------------------------------------------
for (k=1;k<=250;k++)
{
for (l=1;l<=250;l++)
{
Kc[k][l]=0;
}}
//----------------- the beginning of the loop elements -----------------
for(u=1;u<=ne;u++)
{
for(j=1;j<=4;j++)
{
xe[j]=x[ie[u][j]];
ye[j]=y[ie[u][j]];
}
x1=xe[1];x2=xe[2];x3=xe[3];x4=xe[4]; //determining coordinates
of a single element
y1=ye[1];y2=ye[2];y3=ye[3];y4=ye[4];
for (k=1;k<=4;k++)                    //resetting single array Kc_e
{
for (l=1;l<=4;l++)
{
Kc_e[k][l]=0;
}
czlon_lambda_x= (lambdaX/6)*(((y4-y1))/((x2-x1)));
czlon_lambda_y= (lambdaY/6)*(((x2-x1))/((y4-y1)));
Kc_e[1][1]=(czlon_lambda_x)*2+(czlon_lambda_y)*2;
Kc_e[1][2]=(czlon_lambda_x)*-2+(czlon_lambda_y)*1;
Kc_e[1][3]=(czlon_lambda_x)*-1+(czlon_lambda_y)*-1;
Kc_e[1][4]=(czlon_lambda_x)*1+(czlon_lambda_y)*-2;
Kc_e[2][1]=(czlon_lambda_x)*-2+(czlon_lambda_y)*1;
Kc_e[2][2]=(czlon_lambda_x)*2+(czlon_lambda_y)*2;
Kc_e[2][3]=(czlon_lambda_x)*1+(czlon_lambda_y)*-2;
Kc_e[2][4]=(czlon_lambda_x)*-1+(czlon_lambda_y)*-1;
Kc_e[3][1]=(czlon_lambda_x)*-1+(czlon_lambda_y)*-1;
Kc_e[3][2]=(czlon_lambda_x)*1+(czlon_lambda_y)*-2;
Kc_e[3][3]=(czlon_lambda_x)*2+(czlon_lambda_y)*2;
Kc_e[3][4]=(czlon_lambda_x)*-2+(czlon_lambda_y)*1;
Kc_e[4][1]=(czlon_lambda_x)*1+(czlon_lambda_y)*-2;
```

```
Kc_e[4][2]=(czlon_lambda_x)*-1+(czlon_lambda_y)*-1;
Kc_e[4][3]=(czlon_lambda_x)*-2+(czlon_lambda_y)*1;
Kc_e[4][4]=(czlon_lambda_x)*2+(czlon_lambda_y)*2;
std::cout<<std::endl;
for(i=1;i<=4;i++)                //the creation of a global matrix Kc
{
ii=ie[u][i];
for(j=1;j<=4;j++)
{
jj=ie[u][j];
if(jj>0)
Kc[ii][jj]=Kc[ii][jj]+Kc_e[i][j];
}}}}
//--------------------- the end of the loop elements ---------------------
//Calculation of matrix thermal conductivity taking into account
the convective boundary
conditions Kalfa_e for single element and  global matrix Kalfa
//---------------------------------------------------------------------------
for (k=1;k<=250;k++)
{
for (l=1;l<=250;l++)
{
Kalfa[k][l]=0;
}}
//----------------- the beginning of the loop elements -----------------
for(u=1;u<=ne;u++)
{
for(j=1;j<=4;j++)
{
xe[j]=x[ie[u][j]];
ye[j]=y[ie[u][j]];
}
x1=xe[1];x2=xe[2];x3=xe[3];x4=xe[4]; //determining coordinates
of a single element
y1=ye[1];y2=ye[2];y3=ye[3];y4=ye[4];
for (k=1;k<=4;k++)            //resetting single array Kalfa_e
{
for (l=1;l<=4;l++)
{
Kalfa_e[k][l]=0;
}}
if (u==21 ||u==22||u==23||u==24||u==25)    //boundary condition,
right section
{
//exchange heat on the side 2-3 according to the local numbering nodes
```

```
czlon_alfa_bok23= (alfa2*(y4-y1))/6;
Kalfa_e[1][1]= 0;
Kalfa_e[1][2]= 0;
Kalfa_e[1][3]= 0;
Kalfa_e[1][4]= 0;
Kalfa_e[2][1]=0;
Kalfa_e[2][2]=czlon_alfa_bok23*2;
Kalfa_e[2][3]=czlon_alfa_bok23*1;
Kalfa_e[2][4]=0;
Kalfa_e[3][1]=0;
Kalfa_e[3][2]=czlon_alfa_bok23*1;
Kalfa_e[3][3]=czlon_alfa_bok23*2;
Kalfa_e[3][4]=0;
Kalfa_e[4][1]=0;
Kalfa_e[4][2]=0;
Kalfa_e[4][3]=0;
Kalfa_e[4][4]=0;
}
if (u==1 ||u==2||u==3||u==4||u==5)        //boundary condition, left
section
{
//exchange heat on the side 4-1 according to the local numbering nodes
czlon_alfa_bok41= (alfa4*(y4-y1))/6;
Kalfa_e[1][1]= czlon_alfa_bok41*2;
Kalfa_e[1][2]= 0;
Kalfa_e[1][3]= 0;
Kalfa_e[1][4]= czlon_alfa_bok41*1;
Kalfa_e[2][1]=0;
Kalfa_e[2][2]=0;
Kalfa_e[2][3]=0;
Kalfa_e[2][4]=0;
Kalfa_e[3][1]=0;
Kalfa_e[3][2]=0;
Kalfa_e[3][3]=0;
Kalfa_e[3][4]=0;
Kalfa_e[4][1]=czlon_alfa_bok41*1;
Kalfa_e[4][2]=0;
Kalfa_e[4][3]=0;
Kalfa_e[4][4]=czlon_alfa_bok41*2;
}
if (u==1 ||u==6||u==11||u==16||u==21)        //boundary condition,
bottom section
{
//exchange heat on the side 1-2 according to the local numbering nodes
czlon_alfa_bok12= (alfa1*(x2-x1))/6;
```

```
Kalfa_e[1][1]= czlon_alfa_bok12*2;
Kalfa_e[1][2]= czlon_alfa_bok12*1;
Kalfa_e[1][3]= 0;
Kalfa_e[1][4]= 0;
Kalfa_e[2][1]=czlon_alfa_bok12*1;
Kalfa_e[2][2]=czlon_alfa_bok12*2;
Kalfa_e[2][3]=0;
Kalfa_e[2][4]=0;
Kalfa_e[3][1]=0;
Kalfa_e[3][2]=0;
Kalfa_e[3][3]=0;
Kalfa_e[3][4]=0;
Kalfa_e[4][1]=0;
Kalfa_e[4][2]=0;
Kalfa_e[4][3]=0;
Kalfa_e[4][4]=0;
}
if (u==5 ||u==10||u==15||u==20||u==25)      //boundary condition,
top section
{
//exchange heat on the side 3-4 according to the local numbering nodes
czlon_alfa_bok34= (alfa3*(x2-x1))/6;
Kalfa_e[1][1]= 0;
Kalfa_e[1][2]= 0;
Kalfa_e[1][3]= 0;
Kalfa_e[1][4]= 0;
Kalfa_e[2][1]=0;
Kalfa_e[2][2]=0;
Kalfa_e[2][3]=0;
Kalfa_e[2][4]=0;
Kalfa_e[3][1]=0;
Kalfa_e[3][2]=0;
Kalfa_e[3][3]=czlon_alfa_bok34*2;
Kalfa_e[3][4]=czlon_alfa_bok34*1;
Kalfa_e[4][1]=0;
Kalfa_e[4][2]=0;
Kalfa_e[4][3]=czlon_alfa_bok34*1;
Kalfa_e[4][4]=czlon_alfa_bok34*2;
}
if (u==7||u==8||u==9||u==12||u==13||u==14||u==17||u==18||u==19)
{
Kalfa_e[1][1]=0;Kalfa_e[1][2]=0;Kalfa_e[1][3]=0;Kalfa_e[1][4]=0;
Kalfa_e[2][1]=0;Kalfa_e[2][2]=0;Kalfa_e[2][3]=0;Kalfa_e[2][4]=0;
Kalfa_e[3][1]=0;Kalfa_e[3][2]=0;Kalfa_e[3][3]=0;Kalfa_e[3][4]=0;
Kalfa_e[4][1]=0;Kalfa_e[4][2]=0;Kalfa_e[4][3]=0;Kalfa_e[4][4]=0;
```

```
}
for(i=1;i<=4;i++)      //the creation of a global matrix Kalfa
{
ii=ie[u][i];
for(j=1;j<=4;j++)
{
jj=ie[u][j];
if(jj>0)
Kalfa[ii][jj]=Kalfa[ii][jj]+Kalfa_e[i][j];
  }}}
//-------------------- the end of the loop elements --------------------
//----------------------------------------------------------------------
//Determination of the final matrix K = Kc+Kalfa
//----------------------------------------------------------------------
for (k=1;k<=nw;k++)           //matrix addition Kc i Kalfa
{
for (l=1;l<=nw;l++)
{
K[k][l]=Kc[k][l]+Kalfa[k][l];
}}
//----------------------------------------------------------------------
//Calculation of the free vector fq_duze_e for single element
//and global vector taking into account
the power sources of heat per unit volume
//----------------------------------------------------------------------
for (k=1;k<=250;k++)           //resetting vector fq_duze
{
for (l=1;l<=2;l++)
{
fq_duze[k][l]=0;
}}
//----------------- the beginning of the loop elements -----------------
for(u=1;u<=ne;u++)
{
for(j=1;j<=4;j++)
{
xe[j]=x[ie[u][j]];
ye[j]=y[ie[u][j]];
}
x1=xe[1];x2=xe[2];x3=xe[3];x4=xe[4];   //determining coordi-
nates of a single element
y1=ye[1];y2=ye[2];y3=ye[3];y4=ye[4];
for (k=1;k<=4;k++)              //resetting vector fq_male_e for sin-
gle element
{
```

```
for (l=1;l<=1;l++)
{
fq_duze_e[k][l]=0;
}}
czlon_fq_duze= ((moc_zrodel)*((x2-x1)*(y4-y1)))/4;
fq_duze_e[1][1]= czlon_fq_duze*1;fq_duze_e[2][1]= czlon_fq_duze*1;
fq_duze_e[3][1]= czlon_fq_duze*1;fq_duze_e[4][1]= czlon_fq_duze*1;
for(i=1;i<=4;i++)      //the creation of a global fq_duze
{
ii=ie[u][i];
if(jj>0)
fq_duze[ii][1]=fq_duze[ii][1]+fq_duze_e[i][1];
}}
//--------------------  the end of the loop elements ---------------------
//-------------------------------------------------------------------------
//Calculation of the free vector falfa_e for single element
//and global vector falfa taking into account convection heat transfer
//-------------------------------------------------------------------------
for (k=1;k<=250;k++)           //resetting vector falfa
{
for (l=1;l<=2;l++)
{
falfa[k][l]=0;
}}
//-----------------  the beginning of the loop elements -----------------
for(u=1;u<=ne;u++)
{
for(j=1;j<=4;j++)
{
xe[j]=x[ie[u][j]];
ye[j]=y[ie[u][j]];
}
x1=xe[1];x2=xe[2];x3=xe[3];x4=xe[4]; //determining coordinates of a sin-
gle element
y1=ye[1];y2=ye[2];y3=ye[3];y4=ye[4];
for (k=1;k<=4;k++)       //resetting vector falfa_e for single element
{
for (l=1;l<=1;l++)
{
falfa_e[k][l]=0;
}}
if (u==21 ||u==22||u==23||u==24||u==25)  //boundary condi-
tion, right section
{
czlon_falfa= ((alfa2*t_czynnika2)*(y4-y1))/2;
```

```
falfa_e[1][1]=0;
falfa_e[2][1]=czlon_falfa*1;
falfa_e[3][1]=czlon_falfa*1;
falfa_e[4][1]=0;
}
if (u==1 ||u==2||u==3||u==4||u==5)     //boundary condition, left section
{
czlon_falfa= ((alfa4*t_czynnika4)*(y4-y1))/2;
falfa_e[1][1]=czlon_falfa*1;
falfa_e[2][1]=0;
falfa_e[3][1]=0;
falfa_e[4][1]=czlon_falfa*1;
}
if (u==1 ||u==6||u==11||u==16||u==21)     //boundary condition, bot-
tom section
{
czlon_falfa= ((alfa1*t_czynnika1)*(x2-x1))/2;
falfa_e[1][1]=czlon_falfa*1;
falfa_e[2][1]=czlon_falfa*1;
falfa_e[3][1]=0;
falfa_e[4][1]=0;
}
if (u==5 ||u==10||u==15||u==20||u==25)     //boundary condition, top sec-
tion
{
czlon_falfa= ((alfa3*t_czynnika3)*(x2-x1))/2;
falfa_e[1][1]=0;
falfa_e[2][1]=0;
falfa_e[3][1]=czlon_falfa*1;
falfa_e[4][1]=czlon_falfa*1;
}
if (u==7 ||u==8||u==9||u==12||u==13||u==14||u==17||u==18||u==19)
{
falfa_e[1][1]=0;falfa_e[2][1]=0;
falfa_e[3][1]=0;falfa_e[4][1]=0;
}
for(i=1;i<=4;i++)      //the creation of a global falfa
{
ii=ie[u][i];
if(jj>0)
falfa[ii][1]=falfa[ii][1]+falfa_e[i][1];
}}
//----------------------the end of the loop elements ----------------------
//---------------------------------------------------------------------
// Determination of the End vector f= falfa+fq_duze
```

```
//---------------------------------------------------------------------
for (k=1;k<=nw;k++)           //adding a vector
{
for (l=1;l<=1;l++)
{
f[k][l]=fq_duze[k][l]+falfa[k][l];
}}
//---------------------------------------------------------------------
// The solution of equations using Gaussian elimination
//---------------------------------------------------------------------
          for(i=0;i<nw;i++)
                    {
                    for(j=0;j<nw;j++)
                    {
                              a[i][j]=a[i][j]+K[i+1][j+1];
                              }
                    a[i][nw]=f[i+1][1];
                    }
          i=-1;
          while (i<nw-1 && spr)
                    {
                    i++;
                    max=a[i][i]*a[i][i];
                    imax=i;
                    for (j=i+1;j<nw;j++)
                              {
                              if (max<a[j][i]*a[j][i])
                                        {
                                        max=a[j][i]*a[j][i];
                                        imax=j;
                                        }
                        &amp;#160;      }
                    if (imax!=i)
                              {
                              for (j=0;j<nw+1;j++)
                                        {
                                        zamiana=a[imax][j];
                                        a[imax][j]=a[i][j];
                                        a[i][j]=zamiana;
                                        }
                              }
                    for(j=i+1;j<nw;j++)
                              {
                              if (!a[i][i]) spr=0;
                              stala=a[j][i]/a[i][i];
```

```
                                for(k=i;k<nw+1;k++) a[j][k]=a[j][k]-sta-
la*a[i][k];
                                }
                    }
                    for (i=nw-1;i>-1;i ------------------------------- )
                    {
                    b=0;
                                for (j=nw-1;j>i;j --- ) b=b+a[i][j]*x_w[j];
                                x_w[i]=(a[i][nw]-b)/a[i][i];
                                if(!a[i][i]) spr=0;
                    }
std::ofstream stream(``Output.txt'') ;
          if (spr)
                    {
                    std::cout<<std::endl;
     std::cout<<``Results''<<std::endl;


                    for (i=0;i<nw;i++)
     {
      std::cout<<``X[''<<i+1<<``] = ''<<x_w[i]<<std::endl;
      stream<<1000*x[i+1]<<``\t''<<1000*y[i+1]<<``\t''<<x_w[i]<<std::
endl;
      }
                    }
          else
                    {
          std::cout<<`` Method Gaussian used in the algorithm solver '';
          std::cout<<``\n it does not give an unambiguous solu-
tion in this case '';
          std::cout<<``\n or the equations does not give an unambigu-
ous solution \n'';
                    }
getchar();
}
```

# Appendix D
# Subroutine: Gauss Method

```
PROGRAM gauss_program
IMPLICIT NONE
INTEGER, PARAMETER :: rozmiar = 3
DOUBLE PRECISION, DIMENSION(rozmiar,rozmiar) :: A
DOUBLE PRECISION, DIMENSION(rozmiar) :: b
! Predefined matrix A
A(1,1)=-1.0
A(1,2)=2.0
A(1,3)=1.0
A(2,1)=1.0
A(2,2)=-3.0
A(2,3)=-2.0
A(3,1)=3.0
A(3,2)=-1.0
A(3,3)=-1.0
! Predefined vector b
b(1)=-1
b(2)=-1
b(3)=4
! Call Gauss method
CALL Gauss_method(A,b,rozmiar)
END PROGRAM gauss_program


SUBROUTINE Gauss_method(A,b,n)
INTEGER :: n, i_max, Maks_bezw_calk
DOUBLE PRECISION, DIMENSION(n,n) :: A, AMatrix
DOUBLE PRECISION, DIMENSION(n) :: b, bVector, xVector
AMatrix = A
bVector = b
```



M. Hojny, *Modeling Steel Deformation in the Semi-Solid State*,
Advanced Structured Materials 47, DOI 10.1007/978-3-319-67976-1

```
xVector(:) = -1
do i = 1, n
if (AMatrix(i,i) == 0) then
i_max = Maks_bezw_calk(AMatrix(i:n,i),(n-i+1)) + i - 1
if(AMatrix(i_max,i) /= 0) then
CALL Podmien_wiersz(i, i_max, AMatrix, n)
CALL Podmien_elem_vec(i, i_max, bVector, n)
     else
       if (AMatrix(Maks_bezw_calk(AMatrix(:,i),n),i) /= 0) then
         STOP 'Impossible to calculate the roots'
       else
         xVector(i) = 0
       end if
end if
end if
  if (AMatrix(i,i) /= 0) then
    do j = (i+1), n
      do k = (i+1), n
        AMatrix(j,k) = AMatrix(j,k) - AMatrix(i,k) * ( AMatrix(j,
i) / AMatrix(i,i) )
      end do
      bVector(j) = bVector(j) - bVector(i) * ( AMatrix(j,i) / AMatrix(i,
i) )
      AMatrix(j,i) = 0
    end do
  end if
      end do
do i = n, 1, -1
  if(xVector(i) /= 0) then
    xVector(i) = bVector(i) - SUM( xVector((i+1):n) * AMatrix(i,(i+1):
n) )
    if(AMatrix(i,i) /= 0) xVector(i) = xVector(i) / AMatrix(i,i)
  end if
end do
print*, 'xVector='
print*, xVector
END SUBROUTINE Gauss_method

SUBROUTINE Podmien_wiersz(r1, r2, A, n)

INTEGER :: r1, r2, n
DOUBLE PRECISION, DIMENSION(n,n) :: A
do i = 1, n
CALL Zamiana( A(r1,i) , A(r2,i) )
end do
```

```
END SUBROUTINE Podmien_wiersz
SUBROUTINE Podmien_elem_vec(i1, i2, A, n)
INTEGER :: n, i1, i2
DOUBLE PRECISION :: swap
DOUBLE PRECISION, DIMENSION(n) :: A
swap = A(i1)
A(i1) = A(i2)
A(i2) = swap
END SUBROUTINE Podmien_elem_vec
FUNCTION Maks_bezw_calk(A, n)
INTEGER :: Maks_bezw_calk, n
DOUBLE PRECISION, DIMENSION(n) :: A
Maks_bezw_calk = 1
do i = 2, n
if(ABS(A(i)) > ABS(A(Maks_bezw_calk))) Maks_bezw_calk = i
end do
END FUNCTION Maks_bezw_calk
SUBROUTINE Zamiana(v1, v2)
DOUBLE PRECISION :: v1, v2, swap
swap = v1
v1 = v2
v2 = swap
END SUBROUTINE Zamiana
```

# Appendix E
# Subroutine: Transformation and Integration (3D)

```
subroutine TRANSFORM_INTEGRATION
use globalvariables !in this module define global variables
implicit REAL*8 (A-H,O-Z)
integer(kind=8), dimension( 8) :: node_local


!WFACTOR - Weighting factor (each Gaussian point)
!COORD - Coordinates
!PKSI - Derivative of shape function by KSI(each Gaussian point)
!PETA - Derivative of shape function by ETA(each Gaussian point)
!PZETA - Derivative of shape function by ZETA(each Gaussian point)
!NO_ELEMENTS - Number of elements in solution domain
!LAMBDA - Thermal conductivity factor W/mK
!DPX - Derivative of shape function by X(each Gaussian point)
!DPY - Derivative of shape function by Y(each Gaussian point)
!DPZ - Derivative of shape function by Z(each Gaussian point)


COORD(1)= -0.5773502692D+00
COORD(2)= +0.5773502692D+00
WFACTOR(1)= +1.0000000000D+00
WFACTOR(2)= +1.0000000000D+00
do kpnt= 1, 2
do jpnt= 1, 2
do ipnt= 1, 2
```



M. Hojny, *Modeling Steel Deformation in the Semi-Solid State*,
Advanced Structured Materials 47, DOI 10.1007/978-3-319-67976-1

```
!Shape function
SHAPE_FUN(ipnt,jpnt,kpnt,1)= 0.125 * (1.d0 - COORD(ipnt)) *
&(1.d0 - COORD(jpnt)) * (1.d0 - COORD(kpnt))
SHAPE_FUN(ipnt,jpnt,kpnt,2)= 0.125 * (1.d0 + COORD(ipnt)) *
&(1.d0 - COORD(jpnt)) * (1.d0 - COORD(kpnt))
SHAPE_FUN(ipnt,jpnt,kpnt,3)= 0.125 * (1.d0 + COORD(ipnt)) *
&(1.d0 + COORD(jpnt)) * (1.d0 - COORD(kpnt))
SHAPE_FUN(ipnt,jpnt,kpnt,4)= 0.125 * (1.d0 - COORD(ipnt)) *
&(1.d0 + COORD(jpnt)) * (1.d0 - COORD(kpnt))
SHAPE_FUN(ipnt,jpnt,kpnt,5)= 0.125 * (1.d0 - COORD(ipnt)) *
&(1.d0 - COORD(jpnt)) * (1.d0 + COORD(kpnt))
SHAPE_FUN(ipnt,jpnt,kpnt,6)= 0.125 * (1.d0 + COORD(ipnt)) *
&(1.d0 - COORD(jpnt)) * (1.d0 + COORD(kpnt))
SHAPE_FUN(ipnt,jpnt,kpnt,7)= 0.125 * (1.d0 + COORD(ipnt)) *
&(1.d0 + COORD(jpnt)) * (1.d0 + COORD(kpnt))
SHAPE_FUN(ipnt,jpnt,kpnt,8)= 0.125 * (1.d0 + COORD(ipnt)) *
&(1.d0 + COORD(jpnt)) * (1.d0 + COORD(kpnt))


!Derivative of shape function
PKSI(jpnt,kpnt,1)= - 0.125 * (1.d0 - COORD(jpnt)) *
&(1.d0 - COORD(kpnt))
PKSI(jpnt,kpnt,2)= + 0.125 * (1.d0 - COORD(jpnt)) *
&(1.d0 - COORD(kpnt))
PKSI(jpnt,kpnt,3)= + 0.125 * (1.d0 + COORD(jpnt)) *
&(1.d0 - COORD(kpnt))
PKSI(jpnt,kpnt,4)= - 0.125 * (1.d0 + COORD(jpnt)) *
&(1.d0 - COORD(kpnt))
PKSI(jpnt,kpnt,5)= - 0.125 * (1.d0 - COORD(jpnt)) *
&(1.d0 + COORD(kpnt))
PKSI(jpnt,kpnt,6)= + 0.125 * (1.d0 - COORD(jpnt)) *
&(1.d0 + COORD(kpnt))
PKSI(jpnt,kpnt,7)= + 0.125 * (1.d0 + COORD(jpnt)) *
&(1.d0 + COORD(kpnt))
PKSI(jpnt,kpnt,8)= - 0.125 * (1.d0 + COORD(jpnt)) *
&(1.d0 + COORD(kpnt))
PETA(ipnt,kpnt,1)= - 0.125 * (1.d0 - COORD(ipnt)) *
&(1.d0 - COORD(kpnt))
PETA(ipnt,kpnt,2)= - 0.125 * (1.d0 + COORD(ipnt)) *
&(1.d0 - COORD(kpnt))
PETA(ipnt,kpnt,3)= + 0.125 * (1.d0 + COORD(ipnt)) *
&(1.d0 - COORD(kpnt))
PETA(ipnt,kpnt,4)= + 0.125 * (1.d0 - COORD(ipnt)) *
&(1.d0 - COORD(kpnt))
PETA(ipnt,kpnt,5)= - 0.125 * (1.d0 - COORD(ipnt)) *
```

```
&(1.d0 + COORD(kpnt))
PETA(ipnt,kpnt,6)= - 0.125 * (1.d0 + COORD(ipnt)) *
&(1.d0 + COORD(kpnt))
PETA(ipnt,kpnt,7)= + 0.125 * (1.d0 + COORD(ipnt)) *
&(1.d0 + COORD(kpnt))
PETA(ipnt,kpnt,8)= + 0.125 * (1.d0 - COORD(ipnt)) *
&(1.d0 + COORD(kpnt))
PZETA(ipnt,jpnt,1)= - 0.125 * (1.d0 - COORD(ipnt)) *
&(1.d0 - COORD(jpnt))
PZETA(ipnt,jpnt,2)= - 0.125 * (1.d0 + COORD(ipnt)) *
&(1.d0 - COORD(jpnt))
PZETA(ipnt,jpnt,3)= - 0.125 * (1.d0 + COORD(ipnt)) *
&(1.d0 + COORD(jpnt))
PZETA(ipnt,jpnt,4)= - 0.125 * (1.d0 - COORD(ipnt)) *
&(1.d0 + COORD(jpnt))
PZETA(ipnt,jpnt,5)= + 0.125 * (1.d0 - COORD(ipnt)) *
&(1.d0 - COORD(jpnt))
PZETA(ipnt,jpnt,6)= + 0.125 * (1.d0 + COORD(ipnt)) *
&(1.d0 - COORD(jpnt))
PZETA(ipnt,jpnt,7)= + 0.125 * (1.d0 + COORD(ipnt)) *
&(1.d0 + COORD(jpnt))
PZETA(ipnt,jpnt,8)= + 0.125 * (1.d0 - COORD(ipnt)) *
&(1.d0 + COORD(jpnt))
enddo
enddo
enddo

do i= 1, NO_ELEMENTS
!Global number for each element
node1= CONNECT_MATRIX(i,1)
node2= CONNECT_MATRIX(i,2)
node3= CONNECT_MATRIX(i,3)
node4= CONNECT_MATRIX(i,4)
node5= CONNECT_MATRIX(i,5)
node6= CONNECT_MATRIX(i,6)
node7= CONNECT_MATRIX(i,7)
node8= CONNECT_MATRIX(i,8)


write (*,*) 'Global numbers for element:', i
write (*,*) '-----------------------------------------------------------'
write (*,*) node1
write (*,*) node2
write (*,*) node3
write (*,*) node4
```

```
write (*,*) node5
write (*,*) node6
write (*,*) node7
write (*,*) node8
!XYZ coordinates for each element
X1= XYZ(node1,1)
X2= XYZ(node2,1)
X3= XYZ(node3,1)
X4= XYZ(node4,1)
X5= XYZ(node5,1)
X6= XYZ(node6,1)
X7= XYZ(node7,1)
X8= XYZ(node8,1)
Y1= XYZ(node1,2)
Y2= XYZ(node2,2)
Y3= XYZ(node3,2)
Y4= XYZ(node4,2)
Y5= XYZ(node5,2)
Y6= XYZ(node6,2)
Y7= XYZ(node7,2)
Y8= XYZ(node8,2)
Z1= XYZ(node1,3)
Z2= XYZ(node2,3)
Z3= XYZ(node3,3)
Z4= XYZ(node4,3)
Z5= XYZ(node5,3)
Z6= XYZ(node6,3)
Z7= XYZ(node7,3)
Z8= XYZ(node8,3)


write (*,*) 'Coordinates of nodes of element:', i
write (*,*) '-------------------------------------------------------------'
write (*,*) X1,Y1,Z1
write (*,*) X2,Y2,Z2
write (*,*) X3,Y3,Z3
write (*,*) X4,Y4,Z4
write (*,*) X5,Y5,Z5
write (*,*) X6,Y6,Z6
write (*,*) X7,Y7,Z7
write (*,*) X8,Y8,Z8


call JACOBIAN (iflaga1, iflaga2,detJ, DPX, DPY, DPZ, PKSI, PETA,
& PZETA, X1, X2, X3, X4, X5, X6, X7, X8,
```

```
& Y1, Y2, Y3, Y4, Y5, Y6, Y7, Y8,
& Z1, Z2, Z3, Z4, Z5, Z6, Z7, Z8 )

!Integration
wspij= 0.d0
do iii= 1, 8
do jjj= 1, 8
          do kpn= 1, 2
          do jpn= 1, 2
          do ipn= 1, 2


segment= abs(detJ(ipn,jpn,kpn))*WFACTOR(ipn)*WFACTOR(jpn)*WFACTOR(kpn)


wspij= wspij + LAMBDA *(DPX(ipn,jpn,kpn,iii)*
&DPX(ipn,jpn,kpn,jjj)
&+DPY(ipn,jpn,kpn,iii)*DPY(ipn,jpn,kpn,jjj)+DPZ(ipn,jpn,kpn,iii)*
&DPZ(ipn,jpn,kpn,jjj)) *segment


enddo
enddo
enddo
enddo
enddo
enddo
return
end


subroutine JACOBIAN (iflaga1, iflaga2,detJ, PX, DPY, DPZ, PKSI,
&PETA, PZETA,X1, X2, X3, X4, X5, X6, X7, X8, Y1, Y2, Y3, Y4, Y5,
&Y6, Y7, Y8,Z1, Z2, Z3, Z4, Z5, Z6, Z7, Z8 )


implicit REAL*8 (A-H,O-Z)
dimension detJ(2,2,2)
dimension PKSI(2,2,8), PETA(2,2,8), PZETA(2,2,8)
dimension DPX(2,2,2,8), DPY(2,2,2,8), DPZ(2,2,2,8)
do kpnt= 1, 2
do jpnt= 1, 2
do ipnt= 1, 2
DPX(ipnt,jpnt,kpnt,1)=0.d0
DPX(ipnt,jpnt,kpnt,2)=0.d0
```

```
DPX(ipnt,jpnt,kpnt,3)=0.d0
DPX(ipnt,jpnt,kpnt,4)=0.d0
DPX(ipnt,jpnt,kpnt,5)=0.d0
DPX(ipnt,jpnt,kpnt,6)=0.d0
DPX(ipnt,jpnt,kpnt,7)=0.d0
DPX(ipnt,jpnt,kpnt,8)=0.d0
DPY(ipnt,jpnt,kpnt,1)=0.d0
DPY(ipnt,jpnt,kpnt,2)=0.d0
DPY(ipnt,jpnt,kpnt,3)=0.d0
DPY(ipnt,jpnt,kpnt,4)=0.d0
DPY(ipnt,jpnt,kpnt,5)=0.d0
DPY(ipnt,jpnt,kpnt,6)=0.d0
DPY(ipnt,jpnt,kpnt,7)=0.d0
DPY(ipnt,jpnt,kpnt,8)=0.d0
DPZ(ipnt,jpnt,kpnt,1)=0.d0
DPZ(ipnt,jpnt,kpnt,2)=0.d0
DPZ(ipnt,jpnt,kpnt,3)=0.d0
DPZ(ipnt,jpnt,kpnt,4)=0.d0
DPZ(ipnt,jpnt,kpnt,5)=0.d0
DPZ(ipnt,jpnt,kpnt,6)=0.d0
DPZ(ipnt,jpnt,kpnt,7)=0.d0
DPZ(ipnt,jpnt,kpnt,8)=0.d0

DPXDPKSI =

& + PKSI(jpnt,kpnt,1) * X1 + PKSI(jpnt,kpnt,2) * X2
& + PKSI(jpnt,kpnt,3) * X3 + PKSI(jpnt,kpnt,4) * X4
& + PKSI(jpnt,kpnt,5) * X5 + PKSI(jpnt,kpnt,6) * X6
& + PKSI(jpnt,kpnt,7) * X7 + PKSI(jpnt,kpnt,8) * X8
DPYDPKSI =
& + PKSI(jpnt,kpnt,1) * Y1 + PKSI(jpnt,kpnt,2) * Y2
& + PKSI(jpnt,kpnt,3) * Y3 + PKSI(jpnt,kpnt,4) * Y4
& + PKSI(jpnt,kpnt,5) * Y5 + PKSI(jpnt,kpnt,6) * Y6
& + PKSI(jpnt,kpnt,7) * Y7 + PKSI(jpnt,kpnt,8) * Y8
DPZDPKSI =
& + PKSI(jpnt,kpnt,1) * Z1 + PKSI(jpnt,kpnt,2) * Z2
& + PKSI(jpnt,kpnt,3) * Z3 + PKSI(jpnt,kpnt,4) * Z4
& + PKSI(jpnt,kpnt,5) * Z5 + PKSI(jpnt,kpnt,6) * Z6
& + PKSI(jpnt,kpnt,7) * Z7 + PKSI(jpnt,kpnt,8) * Z8
DPXDPETA =
& + PETA(ipnt,kpnt,1) * X1 + PETA(ipnt,kpnt,2) * X2
& + PETA(ipnt,kpnt,3) * X3 + PETA(ipnt,kpnt,4) * X4
& + PETA(ipnt,kpnt,5) * X5 + PETA(ipnt,kpnt,6) * X6
& + PETA(ipnt,kpnt,7) * X7 + PETA(ipnt,kpnt,8) * X8
DPYDPETA =
```

```
& + PETA(ipnt,kpnt,1) * Y1 + PETA(ipnt,kpnt,2) * Y2
& + PETA(ipnt,kpnt,3) * Y3 + PETA(ipnt,kpnt,4) * Y4
& + PETA(ipnt,kpnt,5) * Y5 + PETA(ipnt,kpnt,6) * Y6
& + PETA(ipnt,kpnt,7) * Y7 + PETA(ipnt,kpnt,8) * Y8
DPZDPETA =
& + PETA(ipnt,kpnt,1) * Z1 + PETA(ipnt,kpnt,2) * Z2
& + PETA(ipnt,kpnt,3) * Z3 + PETA(ipnt,kpnt,4) * Z4
& + PETA(ipnt,kpnt,5) * Z5 + PETA(ipnt,kpnt,6) * Z6
& + PETA(ipnt,kpnt,7) * Z7 + PETA(ipnt,kpnt,8) * Z8
DPXDPZETA =
& + PZETA(ipnt,jpnt,1) * X1 + PZETA(ipnt,jpnt,2) * X2
& + PZETA(ipnt,jpnt,3) * X3 + PZETA(ipnt,jpnt,4) * X4
& + PZETA(ipnt,jpnt,5) * X5 + PZETA(ipnt,jpnt,6) * X6
& + PZETA(ipnt,jpnt,7) * X7 + PZETA(ipnt,jpnt,8) * X8
DPYDPZETA =
& + PZETA(ipnt,jpnt,1) * Y1 + PZETA(ipnt,jpnt,2) * Y2
& + PZETA(ipnt,jpnt,3) * Y3 + PZETA(ipnt,jpnt,4) * Y4
& + PZETA(ipnt,jpnt,5) * Y5 + PZETA(ipnt,jpnt,6) * Y6
& + PZETA(ipnt,jpnt,7) * Y7 + PZETA(ipnt,jpnt,8) * Y8
DPZDPZETA =
& + PZETA(ipnt,jpnt,1) * Z1 + PZETA(ipnt,jpnt,2) * Z2
& + PZETA(ipnt,jpnt,3) * Z3 + PZETA(ipnt,jpnt,4) * Z4
& + PZETA(ipnt,jpnt,5) * Z5 + PZETA(ipnt,jpnt,6) * Z6
& + PZETA(ipnt,jpnt,7) * Z7 + PZETA(ipnt,jpnt,8) * Z8
detJ(ipnt,jpnt,kpnt)= DPXDPKSI*(DPYDPETA*DPZDPZETA-DPZDPETA*
&DPYDPZETA) + DPYDPKSI*(DPZDPETA*DPXDPZETA-DPXDPETA*DPZDPZETA) +
&DPZDPKSI*(DPXDPETA*DPYDPZETA-DPYDPETA*DPXDPZETA)


segment= 1.d0 / detJ(ipnt,jpnt,kpnt)
w11= segment * ( DPYDPETA*DPZDPZETA - DPZDPETA*DPYDPZETA )
w12= segment * ( DPZDPKSI*DPYDPZETA - DPYDPKSI*DPZDPZETA )
w13= segment * ( DPYDPKSI*DPZDPETA - DPZDPKSI*DPYDPETA )
w21= segment * ( DPZDPETA*DPXDPZETA - DPXDPETA*DPZDPZETA )
w22= segment * ( DPXDPKSI*DPZDPZETA - DPZDPKSI*DPXDPZETA )
w23= segment * ( DPZDPKSI*DPXDPETA - DPXDPKSI*DPZDPETA )
w31= segment * ( DPXDPETA*DPYDPZETA - DPYDPETA*DPXDPZETA )
w32= segment * ( DPYDPKSI*DPXDPZETA - DPXDPKSI*DPYDPZETA )
w33= segment * ( DPXDPKSI*DPYDPETA - DPYDPKSI*DPXDPETA )


detJ(ipnt,jpnt,kpnt)= dabs(detJ(ipnt,jpnt,kpnt))


DPX(ipnt,jpnt,kpnt,1)= w11*PKSI(jpnt,kpnt,1) +
&w12*PETA(ipnt,kpnt,1) + w13*PZETA(ipnt,jpnt,1)
```

```
DPX(ipnt,jpnt,kpnt,2)= w11*PKSI(jpnt,kpnt,2) +
&w12*PETA(ipnt,kpnt,2) + w13*PZETA(ipnt,jpnt,2)
DPX(ipnt,jpnt,kpnt,3)= w11*PKSI(jpnt,kpnt,3) +
&w12*PETA(ipnt,kpnt,3) + w13*PZETA(ipnt,jpnt,3)
DPX(ipnt,jpnt,kpnt,4)= w11*PKSI(jpnt,kpnt,4) +
&w12*PETA(ipnt,kpnt,4) + w13*PZETA(ipnt,jpnt,4)
DPX(ipnt,jpnt,kpnt,5)= w11*PKSI(jpnt,kpnt,5) +
&w12*PETA(ipnt,kpnt,5) + w13*PZETA(ipnt,jpnt,5)
DPX(ipnt,jpnt,kpnt,6)= w11*PKSI(jpnt,kpnt,6) +
&w12*PETA(ipnt,kpnt,6) + w13*PZETA(ipnt,jpnt,6)
DPX(ipnt,jpnt,kpnt,7)= w11*PKSI(jpnt,kpnt,7) +
&w12*PETA(ipnt,kpnt,7) + w13*PZETA(ipnt,jpnt,7)
DPX(ipnt,jpnt,kpnt,8)= w11*PKSI(jpnt,kpnt,8) +
&w12*PETA(ipnt,kpnt,8) + w13*PZETA(ipnt,jpnt,8)
DPY(ipnt,jpnt,kpnt,1)= w21*PKSI(jpnt,kpnt,1) +
&w22*PETA(ipnt,kpnt,1) + w23*PZETA(ipnt,jpnt,1)
DPY(ipnt,jpnt,kpnt,2)= w21*PKSI(jpnt,kpnt,2) +
&w22*PETA(ipnt,kpnt,2) + w23*PZETA(ipnt,jpnt,2)
DPY(ipnt,jpnt,kpnt,3)= w21*PKSI(jpnt,kpnt,3) +
&w22*PETA(ipnt,kpnt,3) + w23*PZETA(ipnt,jpnt,3)
DPY(ipnt,jpnt,kpnt,4)= w21*PKSI(jpnt,kpnt,4) +
&w22*PETA(ipnt,kpnt,4) + w23*PZETA(ipnt,jpnt,4)
DPY(ipnt,jpnt,kpnt,5)= w21*PKSI(jpnt,kpnt,5) +
&w22*PETA(ipnt,kpnt,5) + w23*PZETA(ipnt,jpnt,5)
DPY(ipnt,jpnt,kpnt,6)= w21*PKSI(jpnt,kpnt,6) +
&w22*PETA(ipnt,kpnt,6) + w23*PZETA(ipnt,jpnt,6)
DPY(ipnt,jpnt,kpnt,7)= w21*PKSI(jpnt,kpnt,7) +
&w22*PETA(ipnt,kpnt,7) + w23*PZETA(ipnt,jpnt,7)
DPY(ipnt,jpnt,kpnt,8)= w21*PKSI(jpnt,kpnt,8) +
&w22*PETA(ipnt,kpnt,8) + w23*PZETA(ipnt,jpnt,8)
DPZ(ipnt,jpnt,kpnt,1)= w31*PKSI(jpnt,kpnt,1) +
&w32*PETA(ipnt,kpnt,1) + w33*PZETA(ipnt,jpnt,1)
DPZ(ipnt,jpnt,kpnt,2)= w31*PKSI(jpnt,kpnt,2) +
&w32*PETA(ipnt,kpnt,2) + w33*PZETA(ipnt,jpnt,2)
DPZ(ipnt,jpnt,kpnt,3)= w31*PKSI(jpnt,kpnt,3) +
&w32*PETA(ipnt,kpnt,3) + w33*PZETA(ipnt,jpnt,3)
DPZ(ipnt,jpnt,kpnt,4)= w31*PKSI(jpnt,kpnt,4) +
&w32*PETA(ipnt,kpnt,4) + w33*PZETA(ipnt,jpnt,4)
DPZ(ipnt,jpnt,kpnt,5)= w31*PKSI(jpnt,kpnt,5) +
&w32*PETA(ipnt,kpnt,5) + w33*PZETA(ipnt,jpnt,5)
DPZ(ipnt,jpnt,kpnt,6)= w31*PKSI(jpnt,kpnt,6) +
&w32*PETA(ipnt,kpnt,6) + w33*PZETA(ipnt,jpnt,6)
DPZ(ipnt,jpnt,kpnt,7)= w31*PKSI(jpnt,kpnt,7) +
&w32*PETA(ipnt,kpnt,7) + w33*PZETA(ipnt,jpnt,7)
```

```
DPZ(ipnt,jpnt,kpnt,8)= w31*PKSI(jpnt,kpnt,8) +
&w32*PETA(ipnt,kpnt,8) + w33*PZETA(ipnt,jpnt,8)
enddo
enddo
enddo
return
end
```

# Appendix F
# Function Calculating Geometry for Stereo Presentation

```
void Stereo
(float x_min,float x_max,float y_min,float y_max,float z_near,float z_far,
float z_zp,float dist,float eye)
{
//Determination of the position of the clipping planes
x_min=-6.0;
x_max=6.0;
y_min=-4.8;
y_max=4.8;
z_near=6.0;
z_far=-20.0;
//The position of the plane of zero parallax
z_zp=0.0;
//The distance between the projection center and the parallax plane
dist=10.5;
//Half the distance between the eyes, positive to the
right eye, negative to the left eye
eye=-0.3;
float  dx = x_max - x_min;
float  dy = y_max - y_min;
float  xmid = (x_max + x_min) / 2.0;
float  ymid = (y_max + y_min) / 2.0;
float  clip_near = dist + z_zp - z_near;
float  clip_far  = dist + z_zp - z_far;
float n_over_d = clip_near / dist;
float  topw = n_over_d * dy / 2.0;
float  bottomw = -topw;
float  rightw = n_over_d * (dx / 2.0 - eye);
```



M. Hojny, *Modeling Steel Deformation in the Semi-Solid State*,
Advanced Structured Materials 47, DOI 10.1007/978-3-319-67976-1

```
float  leftw = n_over_d *(-dx / 2.0 - eye);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
glFrustum(leftw, rightw, bottomw, topw, clip_near, clip_far);
glTranslatef(-xmid - eye, -ymid, -z_zp -dist);
}
```